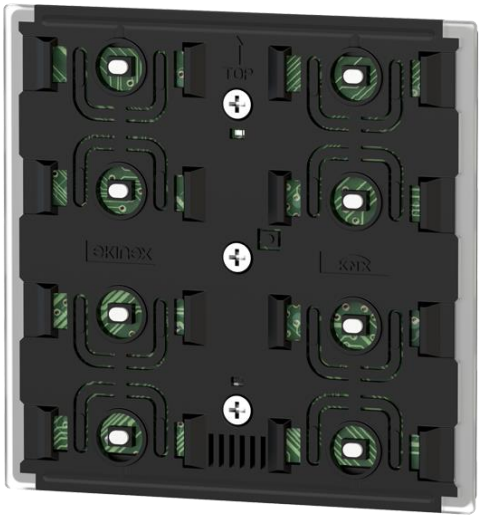
ekinex

CONTROL YOUR LIVING SPACE



Application manual
Wall mounting KNX Pushbuttons with clima controller

EK-ED2-TP 2-4 rockers Series ‘FF
EK-E13-TP 1-4 rockers Series ’71

Summary

Revision	Modifications	Date
4.3	Modification about the key sequence to enter the programming mode	23/05/2023
4.2	Modifications after elimination of the programming button and LED	11/04/2023
4.1	Boundary values update for 2-bytes floating DPT (par. 7.5.3)	13/02/2023
4.0	Upgrade ventilation control, humidification and dehumidification control with relative humidity sensor from bus	20/07/2022
3.0	Modified from E12 to E13	31/05/2022
2.3	Added note for plastic screws mounting (page 7)	04/05/2022
2.2.0	ED2 image updated to 2020 version	04/12/2020
2.1.0	Deleted all references to light sensor	09/03/2020
2.0.0	Updating of the internal temperature controller with related structure of c.o. database.	16/05/2017
1.1.0	Updating of the c.o. indexes in internal database.	30/01/2017
1.0.4	Adding of cross reference to the versions with BG (blue-green) and RW (red-white) LED	04/09/2015
1.0.3	Adding of logic functions. In the 4 rectuangular rockers configuration, it was reintroduced the possibility to control each rocker as a unique rocker, with function A in parallel to function B.	28/08/2015
1.0.2	Updating of temperature control, continuous regulator.	04/08/2015
1.0.1	First emission.	04/05/2015

1 Scope of the document

This application manual describes application details for ekinex® pushbutton interface EK-ED2-TP (2-4 rockers) and for ekinex® pushbutton interface EK-E13-TP (1-4 rockers).

The document is aimed at the system configurator as a description and reference of device features and application programming. For installation, mechanical and electrical details of the device please refer to the technical description datasheet.

Application manual and application programs for ETS are available for download at www.ekinex.com.

Item	File name (## = release)	Version	Device rel.	Update
Product datasheet	STEKED2E32TP_EN.pdf	EK-ED2-TP	A2.0	05 / 2023
Application manual	MAEKED2E13TP_EN.pdf	EK-ED2-TP		
Application program	APEKED2TP##.knxprod	EK-ED2-TP		

Item	File name (## = release)	Version	Device rel.	Update
Product datasheet	STEKE13E23TP_EN .pdf	EK-E13-TP	A2.0	05 / 2023
Application manual	MAEKED2E13TP_EN.pdf	EK-E13-TP		
Application program	APEKE13TP##.knxprod	EK-E13-TP		

You can access the most up-to-date version of the full documentation for the device using following QR codes:

2-4 rocker interface EK-ED2-TP:

1-4 rocker interface EK-E13-TP:

2 Product description

The ekinex® KNX 4-rocker pushbutton unit is a wall-mounting device for on/off switching of loads, dimming of lighting devices, control of motor drives or other programmable switching and control functions. The pushbutton is equipped with an integrated temperature sensor and can act as a room probe or thermostat, both in heating and cooling mode. When acting as a room thermostat, the device is not equipped with a user interface for displaying room conditions and modifying the setpoint temperature; therefore, it must be paired with an external supervision device. Terminals such as radiators, electrical radiators and radiant panels can be controlled.

This unit is equipped with an integrated KNX bus communication module and is designed for wall installation; commands are constituted by rockers with 2 active plus a neutral rest position. The device has also two programmable LEDs for each function which can be used for instance as a status signal or orientation nightlight.

For final use, this unit must be completed with frontal plates for commands, a plastic adapter (for EK-E13-TP model only), a metal support for mounting on a wall-box and a frame. All these items must be ordered separately for mounting the device and in order to obtain the desired aesthetic look; regardless of the detail, several kinds of plates are available (square or rectangular) which can be combined in order to obtain different rockers' combinations.

The device is powered by the KNX bus line with a 30 VDC SELV voltage and does not require auxiliary power.

Product code	Number and type of rockers	Rocker size	Frame
EK-ED2-TP (blue-green led couples) **EK-ED2-TP-RW** (red-white led couples)	2 rectangular vertical 4 square 4 rectangular horizontal	40 x 80 mm 40 x 80 mm 80 x 20 mm	*Form* or *Flank* series
EK-ED2-TP-BG-NF (no frame line, blue-green led couples) **EK-ED2-TP-RW-NF** (no frame line, red-white led couples)			No frame

Product code	Number and type of rockers	Rocker size	Frame
EK-E13-TP-BG (blue-green led couples) **EK-E13-TP-RW** (red-white led couples)	1 single square 2 rectangular vertical 4 square 4 rectangular horizontal	60x60 mm 30x60 mm 30x30 mm 15x60 mm	*Form* or *Flank* series
EK-E13-TP-BG-NF (no frame line, blue-green led couples) **EK-E13-TP-RW-NF** (no frame line, red-white led couples)			No frame

The supply includes, inside the box:

- 2 pairs of fixing screws;
- 1 KNX terminal block for the connection of the bus line.

Mounting information

The screws supplied in the package are suitable for standard installations. For particular applications, in which the screws must be replaced, these must be of the flat-head type.
The supplied plastic screws (# 2) must only be used to hold the push-button panel in position, therefore they must not be tightened with excessive force (max. torque 0.4 Nm).
The screws for the metal support must be tightened with a max. torque of 1.0 Nm.

2.1 Completion of the device

For full installation and operation, the unit must be completed with:

- A rocker faceplate (according to the chosen number and disposition);
- A plastic adapter (for EK-E13-TP version only);
- A metal support for wall-box mounting;
- An ekinex® *Form* or *Flank* series 1-place square or 2-place rectangular frame (with the exception of the rocker interfaces of No Frame series NF);
- An ekinex® 1-window square or 2-window rectangular plate

Rocker placement for EK-ED2-TP

Combining the 3 available models of rockers (rectangular vertical, square horizontal and rectangular horizontal) different configurations are allowed, as shown in the following picture.

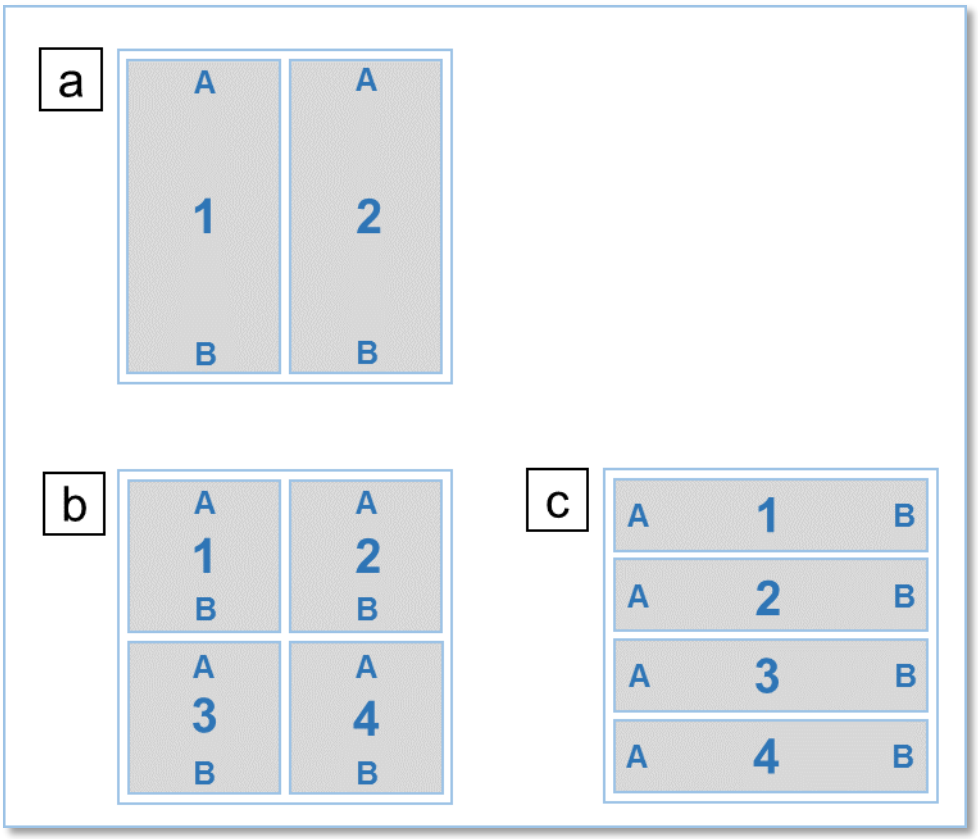


Figure 1A - Rocker combination for EK-ED2-TP

Rocker placement for EK-E13-TP

Combining the 4 available models of rockers (square single, rectangular vertical, square horizontal and rectangular horizontal) different configurations are allowed, as shown in the following picture.

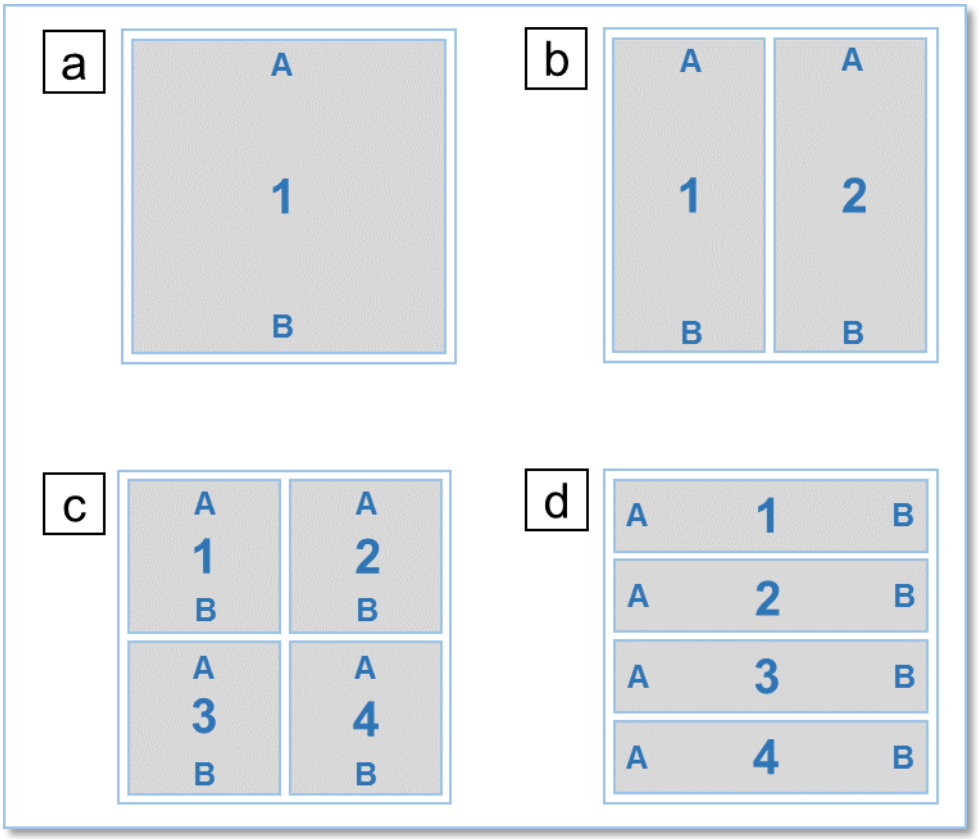


Figure 2B - Rocker combination for EK-E13-TP

2.2 Rocker functions

Each one of two active positions of the rockers (side for rectangular rockers, superior and inferior for square ones) corresponds to an *action*, i.e. an input, or physical pushbutton, of the device. Such actions, in relation to a single rocker, will be labelled with letters A and B.

When one side of a rocker is pressed, the device sends on the KNX bus the telegram (or sequence) associated to the corresponding function according to how the device is programmed.

In the most common situation, for instance, one side of the rocker might send an “ON” telegram for a lighting unit, while the other side would send the “OFF” telegram for the same unit. Another typical application would be for one side of the rocker to increase the brightness of a dimmed light (and respectively decrease it for the opposite side), or to raise / lower a curtain or blind and so on.

The two functions associated with a rocker can also be programmed to perform exactly the same operation, thereby effectively causing one rocker to act as a single pushbutton.

Note for EK-ED2-TP

The use of the entire activation surface of the rocker as if it is a single rocker is programmed by defining the function B as “in parallel with function A as a single function”. This use is configurable only in combinations a), b) and c) (see Figure 1 in the previous page): for instance, it is possible to link only one function to combination a) with a single square rocker.

In combination d) with 4 rectangular horizontal rockers it is possible to link only functions other than function A and function B.

2.3 LED indicators

Each one of the 2 rocker sides is equipped with 4 high efficiency couple of LEDs with different color combinations blue/green and red/white, which can be freely programmed (also with functions not related to the rockers’ functions) e.g. as status feedbacks of the loads or as orientation nightlight.

For a more detailed description of LED position and related settings please refer to the application section of this manual.

2.4 Customization of rocker plates

Rocker plates can be customized with predefined symbols and texts. On request, a customization is also possible with symbols and texts chosen by the customer. For more information see the standard library on the ekinex® catalogue or the website www.ekinex.com.

For further technical information, please also refer to the product datasheet available on the website www.ekinex.com.

3 Switching, display and connection elements

The front side of the device is fitted with mounting hooks for the rocker faceplates; between the hooks there are the pushbuttons and on the lateral sides the LEDs for status indication are placed.

On the rear side, the device is equipped with a programming pushbutton, a programming status LED and terminals for connecting the KNX bus line.

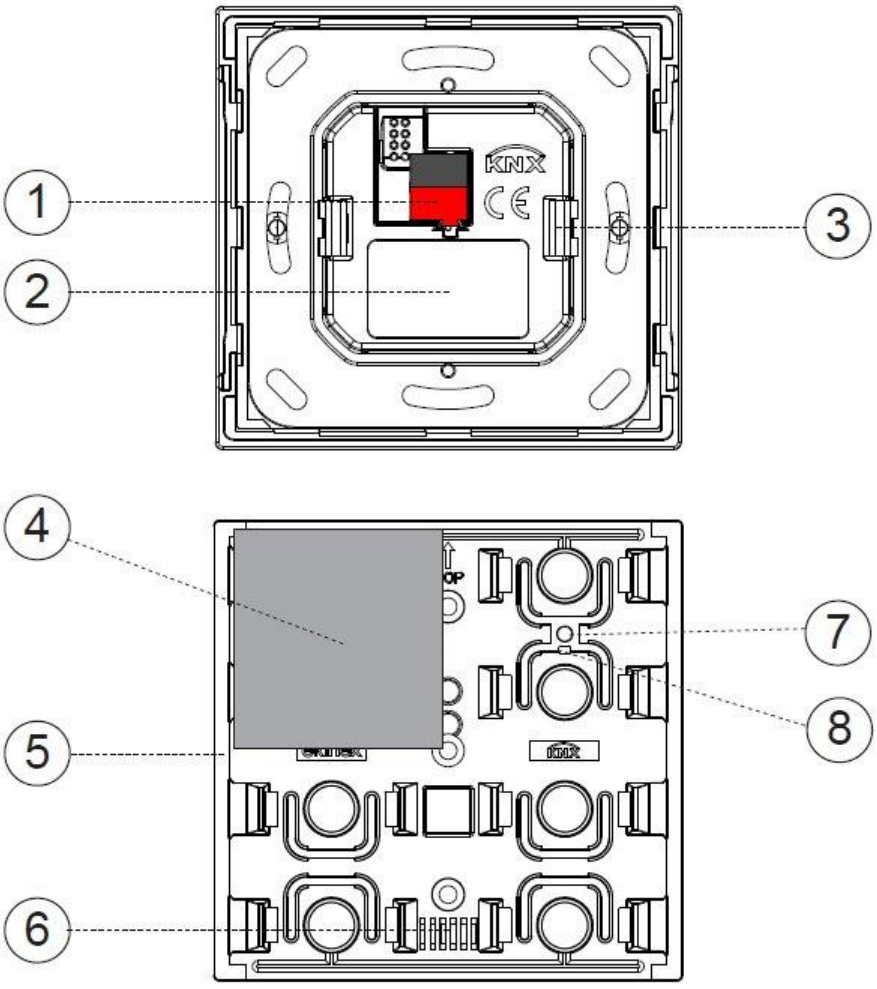


Fig. 2A - Switching, display and connection elements for EK-ED2-TP

1) Connection terminal block for KNX bus line
2) Product label
3) Adapter
4) Rocker (in the example: 30 x 30 mm square type)
5) LED-lightguide
6) Positioning of the temperature sensor
7) Programming pushbutton (in older versions only)
8) Programming LED (in older versions only)

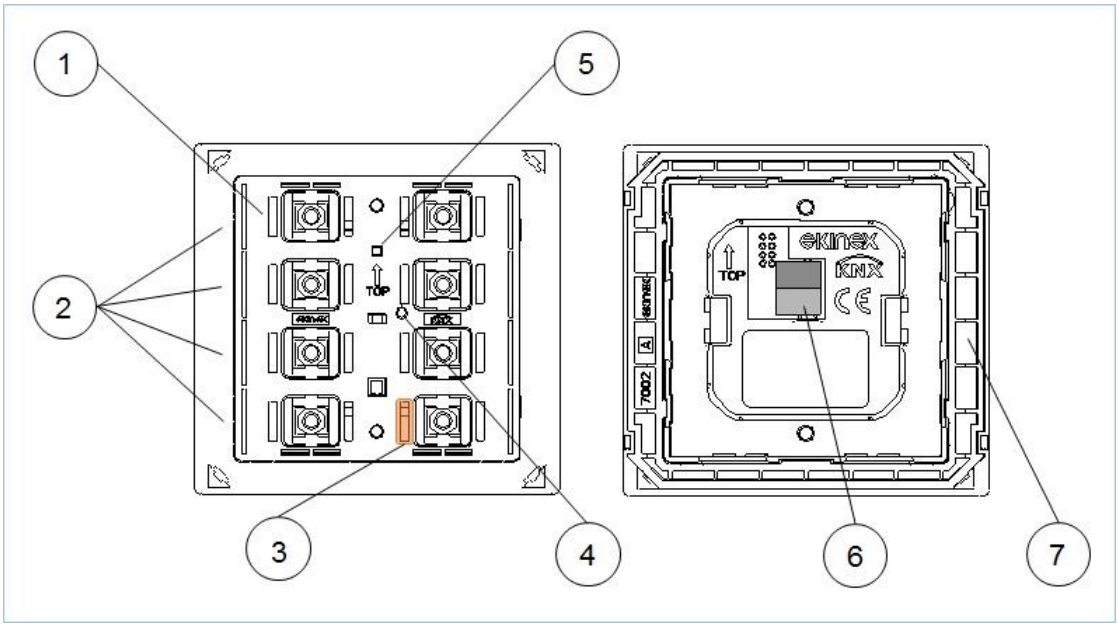


Fig. 2B- Switching, display and connection elements for EK-E13-TP

1. Rocker faceplate hooks
2. LED diffusers
3. Temperature sensor
4. Programming pushbutton (in older versions only)
5. Programming LED (in older versions only)
6. Terminal block for KNX bus line
7. Adapter

4 Configuration

The exact functionality of the device depends on the software settings.

In order to configure and commission the device you need ETS4 or later releases and the ekinex® application program, (namely **APEKED2TPxx.knxprod for EK-ED2-TP and APEKE13TPxx.knxprod for EK-E13-TP**); which can be downloaded from the ekinex website www.ekinex.com.

The application program allows the configuration of all working parameters for the device.

The device-specific application program has to be loaded into ETS or, as alternative, the whole ekinex® product database can be loaded; at this point, all the instances of the selected device type can be added to the project.

For every single device, ETS allows to set the operating parameters separately for each function as described in detail in the following chapters.

The configuration can, and usually will, be performed completely offline; the actual transfer of the programmed configuration to the device takes place in the commissioning phase.

Product code	EAN	No. of channels	ETS application software (## = release)	Communication objects (max no.)	Group addresses (max no.)
EK-ED2-TP		8	APEKED2TP##.knxprod	229	254

Product code	EAN	No. of channels	ETS application software (## = release)	Communication objects (max no.)	Group addresses (max no.)
EK-E13-TP		8	APEKE13TP##.knxprod	229	254

Configuration and commissioning of KNX devices require specialized skills; to acquire these skills, you should attend training courses at a training centre certified by KNX.

For further information: www.knx.org.

5 Commissioning

After the device has been configured within the ETS project according to user requirements, the commissioning of the device requires the following activities:

- electrically connect the device, as described in the product datasheet, to the bus line on the final network or through a purposely setup network for programming;
- apply power to the bus;
- switch the device to programming mode by following the sequence detailed in the next table (both for EK-ED2-TP-… and EK-E13-TP-…);
- upload the configuration (including the physical address) to the device with the ETS program.

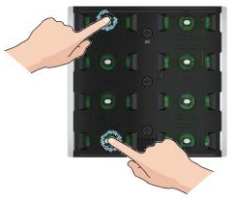
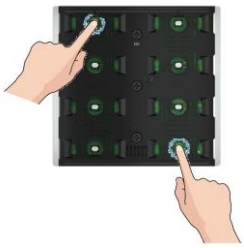
FW version	Programming sequence	Visual feedback
04.xxx e precedenti	*Pressing the programming button*	*Programming LED on steady.*
From 05.xxx to 06.018	 *Simultaneous press of the first and last button on the left side for 5 seconds.*	*All LEDs of the second color flash.* *Programming LED (if present) on steady.*
06.019 and later	 *Simultaneous pressing of the first button on the left side and the last button on the right side for 5 seconds.*	*All LEDs of the second color flash.*

Table 1 - Programming sequences

At the end of the upload, the operation of the device automatically returns to normal mode; the LEDs go out. Now the device is programmed and ready for use on the bus.

6 Function description

After switching on the bus, which also acts as a power supply, the device becomes fully functional after a very short time needed for reinitialization. A delay is programmable for the device to become active on the bus in order to avoid a bus traffic overload during the first moments of start-up of the whole network.

In case of a bus power failure (voltage lower than 19 V for 1 s or more), the device becomes unreactive: before the power supply becomes insufficient, the status is internally stored. The timing functions are not active, neither are the programmed group addresses.

As soon as the bus voltage is restored, the device will resume operation in its previous state (which is saved on power fail), unless different initialization settings are programmed.

6.1 Offline operation

A fully unprogrammed device does not operate in standby mode. Since the operation relies entirely on the exchange of information through communication objects, there is no part of the device that can operate independently from a KNX bus.

6.2 Online operation

In general the device works like a configurable digital sensor that is listening to own inputs or outputs of other devices. On input events the device performs output functionality over KNX bus like sending values or controlling external devices like KNX actuators.

6.3 Software working cycle

The main purpose of the software is following:

- Handle user pushbutton presses and generate bus telegrams according to the assigned functions;
- Implement pushbutton interlock and timing functions;
- Handle incoming bus messages in order to update the status of pushbutton activations and LED indicators;
- Respond to bus messages requesting feedback on the status of the inputs.

The status of the device and specifically of its entities (input activation status and LED indicators) relies on KNX *communication objects*, which can be freely defined and bound in various ways to the physical elements of the device; these communication objects acts as *state variables* for the device.

There are also special events on which it is possible to trigger additional features. These events are the bus failure and recovery, and the download of a new configuration with ETS.

6.4 Pushbutton inputs

The press of a pushbutton can be bound to different effects on a state variable.

6.4.1 Pushbutton input events

A button press can be handled either as an “on-off” event (“on” means when the button is pushed, “off” when it is released), or as a “short press - long press” event (whereby a time period can be defined to discriminate the duration of the “long” from the “short” press).

In both cases, for each of the two available events a separate action can be assigned that operates on a selected variable (actually, more than one; see below for details).

6.4.2 Lock function

For each input (or channel if inputs are coupled, see below), a lock feature can be enabled which allows to block the operation of an input through a message on a communication object.

When in a locked state, the input is effectively disabled.

A value (for each transition) can be specified to be assigned to the communication object upon entering or exiting the locked state.

The locked state can also be automatically activated when the bus is connected.

6.4.3 State variables (communication objects)

The variable that is changed by the input events can be one of the types available for KNX communication objects, i.e. for instance a 1-bit value (on-off), a 2-bit value or an integer value of larger size.

In all cases, each of the two events can:

- change the value of the variable to one of two definable values within its range (which is trivial in the case of the 1-bit value);
- toggle between the two defined values
- do nothing (value is unaffected)

This state variable, once assigned a group address, is actually a **KNX communication object**; as such, it undergoes the usual rules for communication objects, among which – for instance – the effect of flags to determine how the change of value affects the transmission of the objects.

6.4.4 Binding between Events and Communication objects

The above description is a little simplified in order to ease comprehension; as a matter of fact, to each event can be assigned not just one, but several communication objects (up to 8), even of different types. Each of these communication objects can have its own behaviour and its own associated value set.

6.4.5 Repeated send

For most features, is it possible to set the device to send a telegram not just when a value changes as a consequence of an input transition, but also at regular intervals whenever that value setting is active.

This behaviour, also referred to as Cyclical Transmission, can be set separately for each of the two values that are associated to an input (or both, or none of them).

If an input is set to "*send values or sequences*" mode, repeated send is not available if more than 1 Communication Object is assigned to that input.

6.4.6 Input coupling

The 8 pushbutton inputs described can be considered, and used, as independent; however, due to the physical structure of the device and the nature of the functions it most frequently performs, these inputs can be naturally grouped in pairs, which in the application program are referred to as *channels*. Each channel is made of a pair of inputs, and is physically associated to a rocker.

Since the channels of the device are labelled 1 to 4, the inputs are labelled 1A / 1B for channel 1, 2A / 2B for channel 2 and so on. The same numbering is used whether the channel pairing is used or not.

In order to specify channel pairings, each rocker can be configured in two ways: single mode and coupled mode. This setting appears among rocker-level settings rather than input-level settings, because only inputs belonging to the same rocker can be coupled. The only combinations allowed for coupling are in fact 1A with 1B, 2A with 2B, and so on.

- In *single or independent mode*, each input operates independently, has its own parameters and communication objects. This is the mode of operation described so far.
- In *coupled mode*, 2 inputs operate logically grouped under a channel in order to perform a common functionality; therefore, they operate on shared communication objects.

It is possible to configure some of the inputs in *single or independent* mode and the others in *coupled* mode, with the pairing constraints just described.

It must be mentioned that there is actually a third way to configure an input pair, which lies somehow halfway between the two modes above (although it is considered as a variation of the single mode): each second input, i.e. inputs 1B, 2B, 3B etc., can be configured to perform exactly the same function as its first input. In this fashion, both pushbuttons associated with a rocker are effectively operated “in parallel”, so as to operate the whole rocker as a single, larger control (either pushbutton or switch, according to programmed operation).

Following there is a description of all possible features of the channels. *Single or independent* and *coupled* modes have a similar functionality, but differ for the configuration and will be therefore be treated separately.

6.4.7 Single or independent input mode

Each single input can be configured for one of following different features:

1. *Send values or sequences*
   An event triggers the transmission on the bus of configurable values or sequence of values.
   These values can be of a logical type or a numerical type with a different size.
   A sequence of values can be made of up to 8 communication objects of different value types.
   Time delays can set between values in the sequence.
2. *Dimmer control*
   This mode is intended to be used with dimming actuators for the control of lighting devices.
   The functionality is triggered on short press and long press events.
   On short press events, the device sends on/off telegrams to the dimming actuator.
   On long press events, the dimming percentage is varied up or down until the button is released.
3. *Shutter or Venetian blind control*
   This mode is intended to be used together with actuators for the control of motorized blinds, shutters and similar devices. These actuators have functions for blind opening and closing; two movement types are selectable, i.e. continuous movement and stepwise movement. On input events, the device sends operation telegrams to the actuators.
   The operation is configurable through following parameters:
   - If *toggle* mode is enabled, on each activation of the same input the movement direction is inverted; if it is disabled, the movement direction is fixed and it can be set to “up” or “down”.
   - If *blinds* mode is enabled, the device sends “full movement” telegrams on long press and “step” telegrams on short press; if it is disabled, the device sends “full movement” telegrams on long press and “stop” telegrams on short press.
4. *Scene function output*
   This mode is intended to be used together with several KNX actuators that support using a scene function; this function allows storing and recalling a communication object value on an actuator.
   In this mode, the role of the device is to send a “store / recall scene” telegram to the actuator on a long / short press event.
   This mode has two possible configurations:
   - Activate pre-set scene on short press, and store current setting as scene value on long press
   - Activate two different scenes on long and short press.

6.4.8 Coupled input mode

Each pair of coupled inputs, corresponding to the two sides of a same rocker, can be configured for one of following different features (only the differences from the single mode are highlighted):

1. *Switch control*
   Both inputs in a pair are bound to the same communication object; unlike single mode, the object can only be of the 1-bit type (on-off), therefore building a conventional switching behaviour.
   The user can configure which of the two inputs sets the “off” or resp. “on” value.
2. *Dimmer control*
   The functionality is triggered on short press and long press events of the inputs in the pair.
   The user can configure which of the two inputs sets the “up” or resp. “down” value.
   On short press events, the input configured as “up” sends an “on” switching telegram to the dimming actuator; while the “down” input sends an “off” telegram.
   On long press events, the dimming percentage is varied up or down until the button is released.
3. *Shutter or Venetian blind control*
   The two inputs of a pair are assigned to opposite movement directions; these can be assigned to inputs as desired, i.e. A up / B down or the other way around.
   The *blinds* mode can also be set, and it works exactly as in single mode.

In coupled mode, there is no provision for a *scene* control feature.

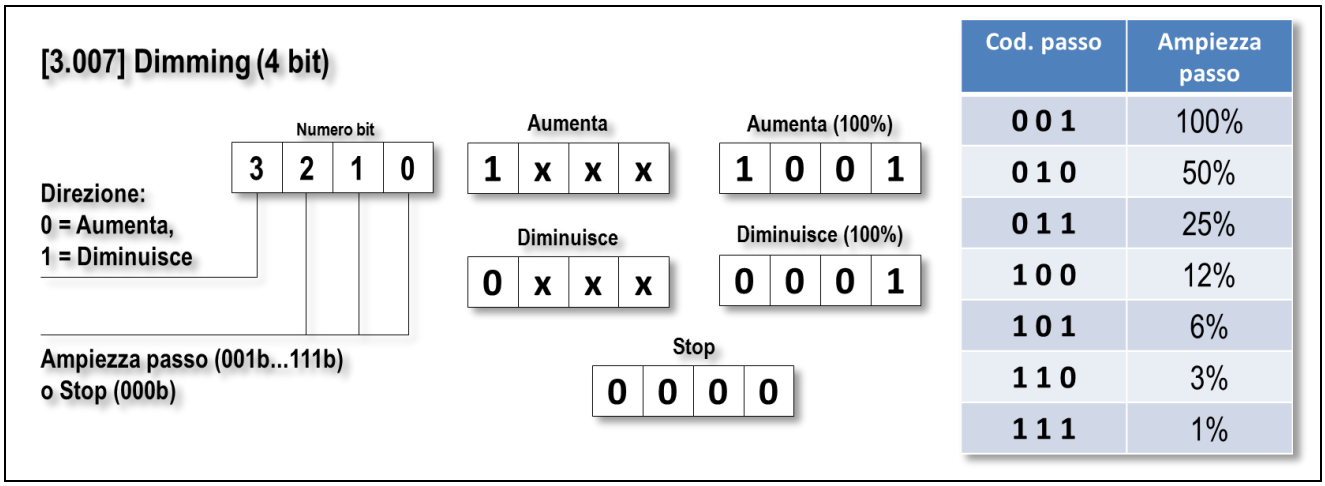
6.4.9 Dimming function

The dimming function is a device application profile included in KNX specifics. Those specifics define the basic requirements for interface mechanisms, in addition to which some aspects regarding the operating modes, peculiar for each device (for both command or actuation devices) are to be considered.

All the information contained in this section have the purpose of illustrating specific device functions, and therefore are not to be necessarily considered exhaustive or applicable to other cases. In order to obtain a complete or generally applicable documentation, please refer to the official KNX documentation.

For further information, visit the website www.knx.org.

The dimmer control type is essentially based on a 4-bit communication object, whose data has the following format:



Cod. passo	Ampiezza passo
0 0 1	100%
0 1 0	50%
0 1 1	25%
1 0 0	12%
1 0 1	6%
1 1 0	3%
1 1 1	1%

The transmission of telegrams containing data of such format tells the actuator to perform an increase or a decrease, by an amplitude equal to the specified step, or to stop an ongoing variation.

The increase or decrease of an intensity value by the actuator is not instantaneous but gradual; therefore, an increase / decrease command with interval equal to the maximum allowed value has the effect of starting the intensity variation in the desired direction, which will continue until the maximum (or minimum) value has been reached. Such variation can be stopped, once the desired intensity value has been reached, by sending a “stop” command.

It is normally possible, and desirable, to have the possibility to instantly switch on or off the load (i.e. to instantaneously bring its value from 0% to 100%). In order to achieve that, an “On / Off” command based on another object is used; this is the same object used for the normal load switch, which is present also in absence of a dimming mechanism.

The command device – in this case, the rocker unit – will define the operations to generate a sequence of commands with an opportune order and time interval, in order to achieve the desired command effect.

In case of unit EK-Ex2-TP, the defined operations and related commands are the following:

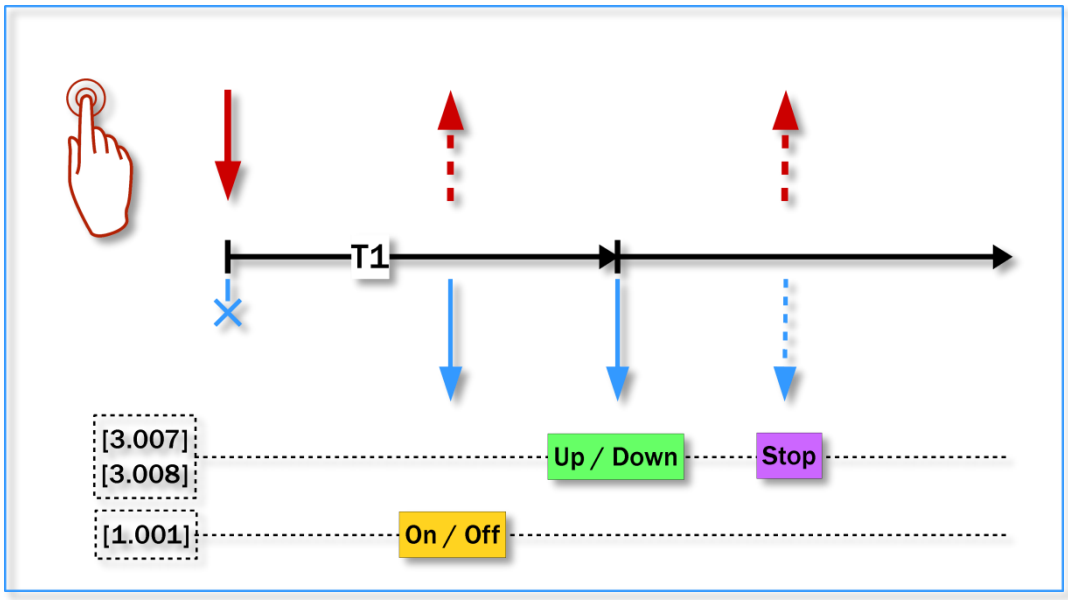


Figure 3 - Dimmer mode command sequence

- Short press: instantaneous switch on / off (toggle on / off on a switch object);
- Long press: increase / decrease value until 100% / 0%;
- Release: stop increase / decrease.

Please note that the same mechanism can be applied to the shutter or venetian blind control (in that case, “maximum / minimum” is substituted with “open / close”). For this purpose, the data type (DPT) 3.008 exists, whose structure and values are identical to those already described; in order to control a shutter with the same mode, it is possible to connect a communication object type 3.007 command side, to an object type 3.008 actuator side (if foreseen). In this case, obviously, the object type “On / Off” which allows instantaneous switch on / off is not used.

6.4.10 Shutter / venetian blind function

The “Shutter / venetian blind” function is a bundle of application profiles included in KNX specifics. As for dimming function, such specifics define basic requirements related to interface mechanisms, in addition to which some aspects regarding the operating modes, peculiar for each device (for both command or actuation devices) are to be considered.

All the information contained in this section have the purpose of illustrating specific device functions, and therefore are not to be necessarily considered exhaustive or applicable to other cases. In order to obtain a complete or generally applicable documentation, please refer to the official KNX documentation.

For further information, visit the website www.knx.org.

In case of shutters, the actuator brings a mechanic component from one point to another in a gradual way, with possibility to stop at intermediate points; the command is carried out by 2 lines which, when activated (one line at a time) make the actuator move in the corresponding direction.

A venetian blind is essentially a shutter that, in addition to the up / down movement, is also equipped with slats that can be opened / closed same way as a shutter (gradual movement between extreme points). The peculiarity is that normally the slat’s movement and the up / down movement are controlled by the same two lines; therefore, the activation of the electromechanic device must be carried out according to a specific sequence. For further detail please check the actuator’s documentation; in this document all we need to point out is that, command side, the control sequences can be considered as independent from these aspects.

The basic control for a shutter or a venetian blind is essentially based on three 1-bit communication objects:

- [1.008] Move Up/Down
- [1.007] Stop – Step Up/Down
- [1.017] Dedicated Stop

The effect of the commands linked to these objects is the following:

- The command “Move”, when received, starts the movement of the shutter in the indicated direction.
- The command “Stop – Step” has two functions: if the shutter is stopped, it moves by one step in the indicated direction (the duration is set in the actuator), if not, it stops the ongoing movement without doing anything else.
- The command “Stop” just stops the ongoing movement.

In addition, other types of control objects are normally available (“dimmer” type, absolute position, etc.) but they are not part of the basic control on which this manual is about; for further information please refer to the actuators’ manual or KNX specifics.

In the simplest version, on command side:

- In order to control a shutter at least the objects “Move” and “Stop” are required (and present).
- In order to control a venetian blind at least the objects “Move” and “Stop – Step” are required (and present).

On actuator side – whether it is a shutter or a venetian blind – the presence of objects “Move” and “Stop – Step” must be guaranteed, while the presence of the object “Stop” is optional (but usually present).

As for the operations to perform on the command device, in our specific case the rocker unit, in order to generate a sequence of these commands with the proper order and time interval, there are multiple possibilities.

In case of ekinex input devices, two modes are available – indicated as “Shutter” and “Venetian blind” based on their typical destination – which are illustrated in the following figure.

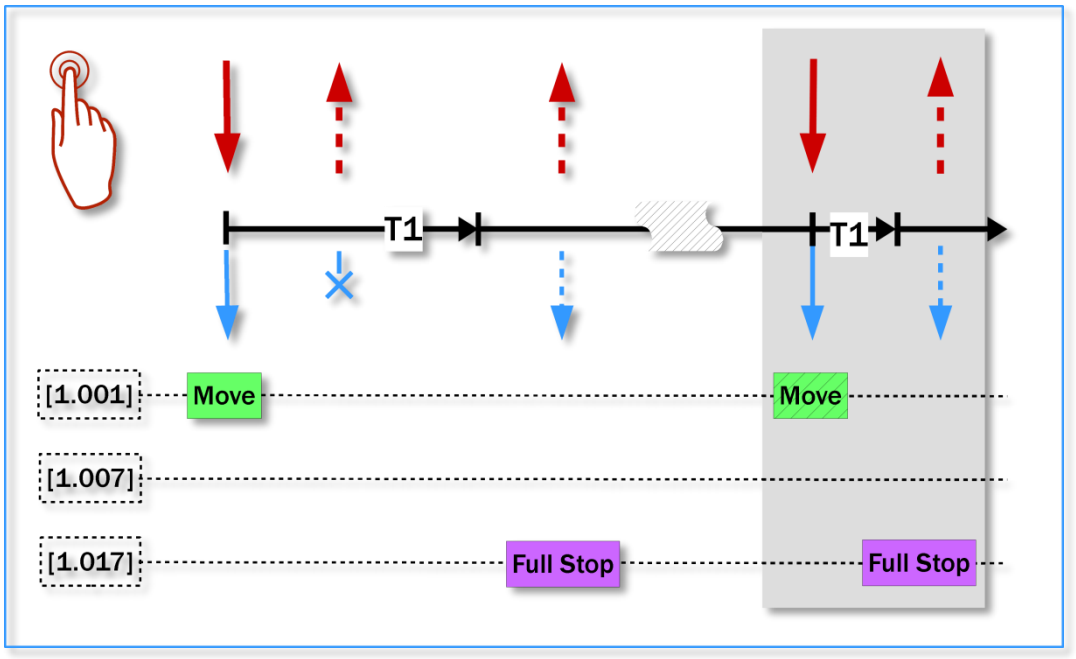


Figure 4 - “Shutter” mode command sequence

In “Shutter” mode, when a rocker is pressed – or a digital input is activated – the shutter starts moving in the corresponding direction (which can be alternatively in the two directions if the rocker is in independent mode and has been configured as *toggle*).

If the rocker is released quickly, the shutter will continue its run until full opening or closing; it is still possible to stop it by pressing again the rocker with a long press.

If the rocker is pressed with a long press, when it is released – which will be in correspondence with the desired position – the shutter will stop.

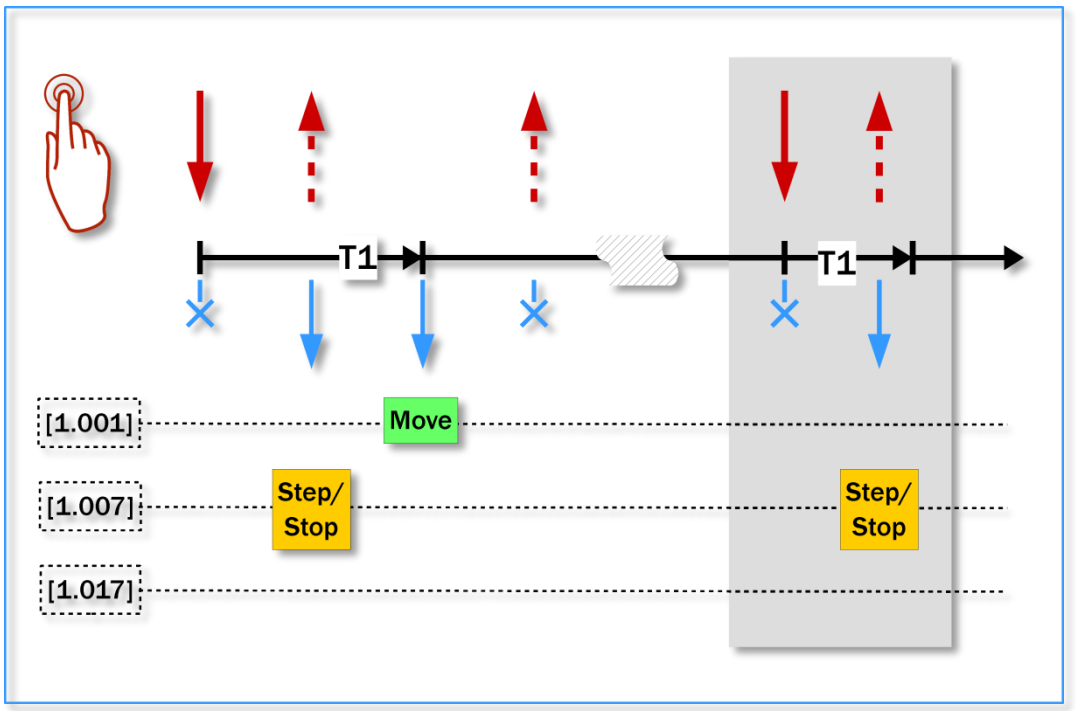


Figure 5 - “Venetian blind” mode command sequence

In “Venetian blind” mode, on release of a rocker after a short press, the venetian blind performs a step; this operation, normally – i.e. even if the actuator is indeed configured for a venetian blind – is used for the slats regulation.

If the rocker is pressed with a long press, when the threshold time is reached, a “Move” command is issued, which will bring the venetian blind to full open or close. In order to stop it at an intermediate position, the rocker needs to be pressed again (short press).

6.5 LED indicators

The LED indicators associated with each input are two (first colour and second colour) and can be singularly addressed, even if the corresponding inputs are coupled.

6.5.1 General parameters

All LEDs have a common intensity value, which can be set from the bus through a communication object or with a fixed setting from 0 to 100% in 10% steps.

6.5.2 Individual parameters

Each LED can be driven in one of following ways:

- Fixed value (always on or always off);
- Lit when the corresponding input is activated. In this option, an additional off-delay can be specified after the button is released;
- Status set from the bus through a communication object. In this case, the LED can be set to be flashing when active (with a choice of different on/off time combinations), and the on/off light status can be inverted with respect to the communication object status (so as to have the LED on when the CO has an “off” value).

6.5.3 Technical Alarm indicator

The device has a peculiar indicator feature called *Technical Alarm*: if it is enabled, all LEDs at the four corners of the device can be activated flashing through a KNX bus telegram. In particular, the activation of the technical alarm generates the blue LEDs activation in BG version (colour of the LEDs: green and blue), while the red LEDs activation in RW version (colour of the LEDs: red and white). For further information about the LEDs position and configuration parameters please refer to the application section of this manual.

This feature is meant as an indicator for a generic alarm condition, but it can be used in a custom way as the user sees fit.

The typical purpose of this indication is to warn about an alarm condition, but can be also used for any other indication.

6.6 Temperature sensor

The value from the embedded temperature sensor, unless it is disabled, can be read from the bus by other devices. In addition, their behaviour can be modified through following parameters:

6.6.1 Temperature sensor

The raw value read from the sensor can be corrected with a small offset (-5 °C to +5 °C in steps of 0.5 °C), in order to compensate for environmental factors and achieve a better precision.
The sensor value can periodically be sent on the bus with a specified transmission interval, and whenever a specified variation occurs.

6.7 Room thermostat

6.7.1 Use of sensors

The temperature controller integrated inside the pushbutton allows the room temperature acquisition in the following ways:

1) from the temperature sensor integrated inside the device;
2) via bus from another KNX device, e.g. another ekinex® pushbutton

In order to optimize or correct the temperature regulation in particular cases (big rooms, when there is a strong asymmetry in temperature distribution, when the pushbutton is installed in wrong or unsuitable positions, etc.) the device can use a weighted mean between two temperature values. The weights are assigned according to the *Relative weight* parameter, which assigns a proportion to the values.

Note on mounting position

If the integrated temperature regulator is used, the device must be preferably installed on an internal wall, at 1,5 m of height and at least 0,3 m of distance from doors. The device cannot be installed near heat sources such as radiators or domestic appliances or in positions subjected to direct solar irradiation. If necessary, for the regulation can be used a weighted mean value between the measured temperature acquired by the integrated sensor and a value received via bus from another KNX device.

6.7.2 Applications

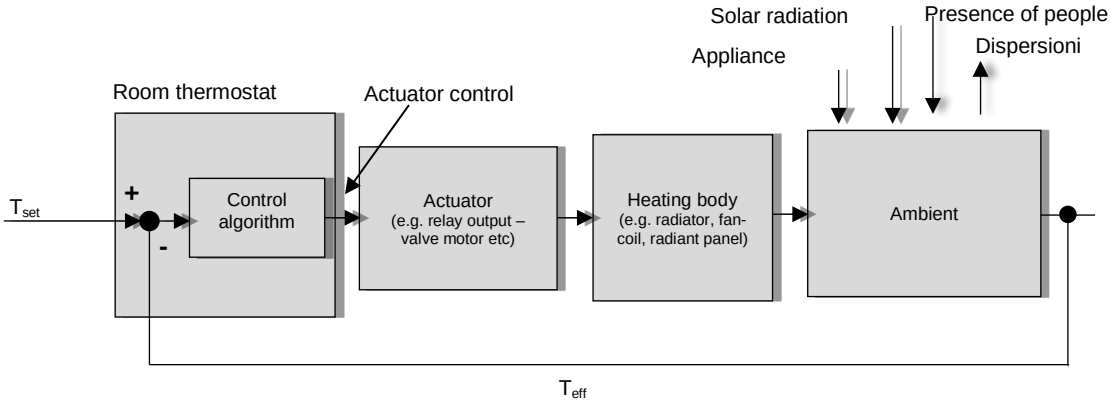
The applications that can be configured are peculiar to thermal plants with a double stage and concern the following terminals: radiators, electric radiators radiant panel systems and fan-coils.

The temperature control can be:

- two point control with hysteresis, ON-OFF command type;
- proportional-integral, with ON-OFF command, PWM or continuous type.

6.7.3 Control algorithms

The picture below shows the components of a common generic control system for ambient temperature. The room thermostat measures the actual temperature of the air mass (T_{eff}) and constantly compares it to the setpoint value (T_{set}).

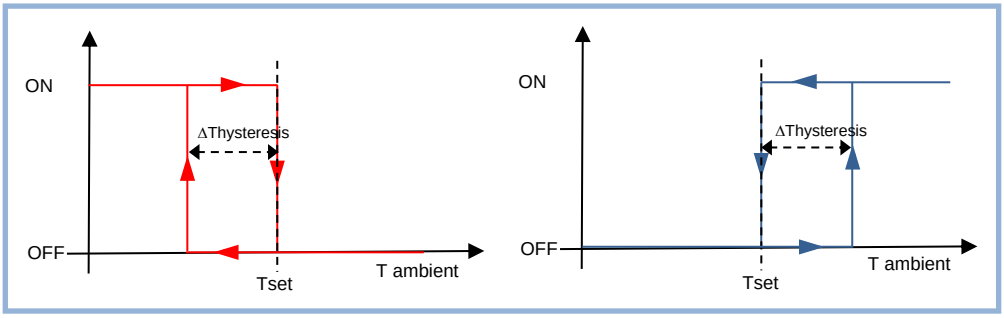


The control algorithm, basing on the difference between T_{set} and T_{eff}, processes a command value which can be analogue or On / Off type; the command is represented by a CO that is transmitted via bus, periodically or event based, to a KNX actuator device.

The output of the actuator device is the driving variable of the control system, which can be e.g. a flow rate of water or air. The control system realized by the room thermostat is of feedback type, namely the algorithm takes into account the effects on the system in order to change the control action on the same entity.

6.7.3.1 Two-point control with hysteresis

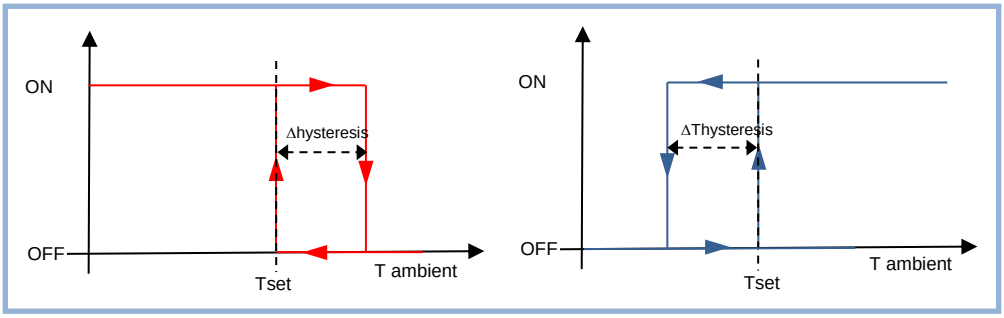
This control algorithm, which is also known as On / Off, is the most classic and popular. The control provides for the on / off switching of the system following a hysteresis loop, i.e. two threshold levels are considered for the switching instead of a single one.



Heating mode: when the measured temperature is lower than the value of the difference ($T_{set} - \Delta T_{hysteresis}$), whereby $\Delta T_{hysteresis}$ identifies the differential adjustment of the boilers, the device activates the heating system by sending a message or KNX telegram to the actuator that handles the heating system; when the measured temperature reaches the desired temperature (`Setpoint`), the device disables the heating system by sending another message. In this way, there are two decision thresholds for activation and deactivation of the heating, the first being the level ($T_{set} - \Delta T_{hysteresis}$) below which the device activates the system, whereas the second is the desired temperature above which the heating system is deactivated.

Cooling mode: When the measured temperature is higher than the value of the difference ($T_{set} + \Delta T_{hysteresis}$), whereby $\Delta T_{hysteresis}$ identifies the differential adjustment of the cooler, the device activates the air conditioning system by sending a message or KNX telegram to the actuator that handles it; when the measured temperature falls below the desired temperature T_{set} the device turns off the air conditioning system by sending another message. In this way, there are two decision thresholds for activation and deactivation of the cooling: the first being the level ($T_{set} + \Delta T_{hysteresis}$) above which the device activates the system, whereas the second is the desired temperature below which the air conditioning system is deactivated. In the ETS application program, two different parameters are available for the hysteresis value for both heating and cooling: the values usually differ depending on the system type and its inertia.

In those applications where floor or ceiling radiant panels are present, it is possible to realize a different 2-point room temperature control. This type of control must be paired either to a proper regulation system for flow temperature that takes into account all internal conditions or an optimizer that exploits the thermal capacity of the building to adjust the energy contributions. In this type of control the hysteresis ($\Delta T_{hysteresis}$) o the room temperature high limit ($T_{set} + \Delta T_{hysteresis}$) represent the maximum level of deviation that the user is willing to accept during plant conduction.



Heating mode – When the measured temperature is lower than the desired temperature T_{set}, the device activates the heating system by sending a message or KNX telegram to the actuator that handles it; when the measured temperature reaches the value (T_{set} + $\Delta T_{hysteresis}$), whereby $\Delta T_{hysteresis}$ identifies the differential adjustment of the boilers the device disables the heating system by sending another message. In this way, there are two decision thresholds for activation and deactivation of the heating, the first being the desired temperature T_{set} below which the device activates the system, whereas the second is the value (T_{set} + $\Delta T_{hysteresis}$), above which the heating system is deactivated.

Cooling mode – When the measured temperature is higher than the desired temperature T_{set}, the device activates the air conditioning system by sending a message or KNX telegram to the actuator that handles it; when the measured temperature reaches the value (T_{set} - $\Delta T_{hysteresis}$), whereby $\Delta T_{hysteresis}$ identifies the differential adjustment of the air conditioning system, the device disables the air conditioning system by sending another message. In this way, there are two decision thresholds for activation and deactivation of the air conditioning system: he first being the desired temperature T_{set} above which the device activates the system, whereas the second is the value (T_{set} - $\Delta T_{hysteresis}$) below which the air conditioning system is deactivated.

In the ETS application program, two different parameters are available for the hysteresis value for both heating and cooling: the values usually differ depending on the system type and its inertia.

In the ETS application program, the default 2-point hysteresis control algorithm foresees inferior hysteresis for heating and superior for cooling. If Heating and/or cooling type = floor radiant panels or ceiling radiant panels, it is possible to select the hysteresis position according to the described second mode, i.e. with superior hysteresis for heating and inferior for cooling.

The desired temperature (T_{set}) is generally different for each one of the 4 operating modes and for heating/cooling modes. The different values are defined for the first time during ETS configuration and can be modified later on. In order to optimize energy saving (for each extra degree of room temperature, outbound dispersions and energy consumption go up 6%), it is possible to take advantage of the multi-functionality of the domotic system, for example with:

- Hour programming with automatic commutation of the operating mode by means of KNX supervisor;
- Automatic commutation of the operating mode according to presence of people in the room;
- Automatic commutation of the operating mode according to window opening for air refreshment;
- Circuit deactivation when desired temperature is reached;
- Flow temperature reduction in case of partial load.

6.7.3.2 Continuous Proportional-Integral control

The continuous proportional-integral (PI) controller is described by the following equation:

$$control\ variable(t) = Kp \times error(t) + Ki \times \int_0^t error(\tau)d\tau$$

whereby:

$error(t) = (Setpoint - Measured\ temperature)\ in\ heating$

$error(t) = (Measured\ temperature - Setpoint)\ in\ cooling$

$Kp = proportional\ constant$

$Ki = intgral\ constant$

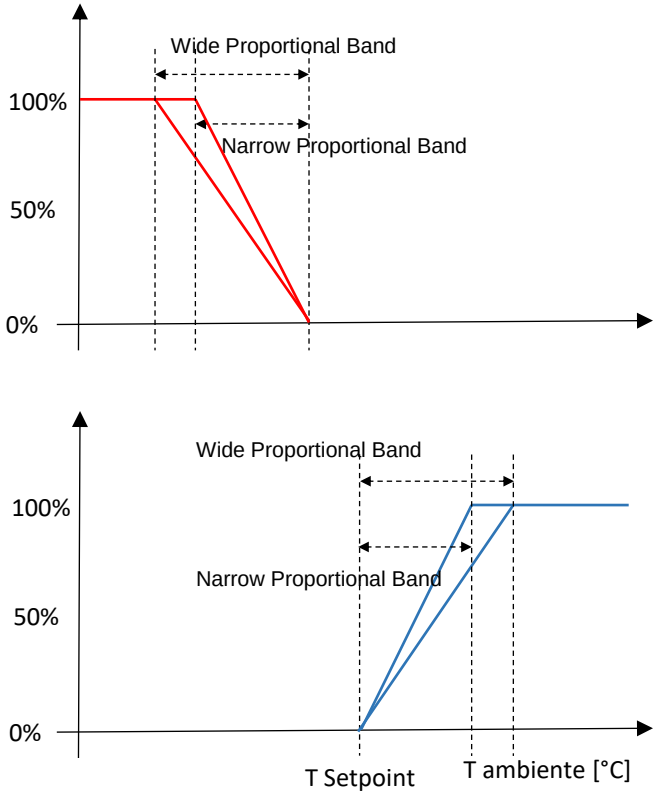
The control variable is composed by 2 numbers, one depending proportionally from the error and one depending from the integral of the error itself.

Practically, some more intuitive values are used:

$$Proportional\ Band\ BP\ [K] = \frac{100}{Kp}$$

$$Integral\ Time\ Ti\ [min] = \frac{Kp}{Ki}$$

The Proportional Band is the error value that determines the maximum span of the control variable at 100%.
For example, a controller with Proportional Band = 5 K regulates at 100% when Setpoint = 20°C and Measured Temperature is ≤ 15 °C in heating mode; in cooling mode, it regulates at 100% when Setpoint = 24°C and Measured Temperature is ≥ 29°C. As shown in figure, a controller with a narrow Proportional Band provides higher control variable values for smaller errors compared to a controller with a wider Proportional Band.

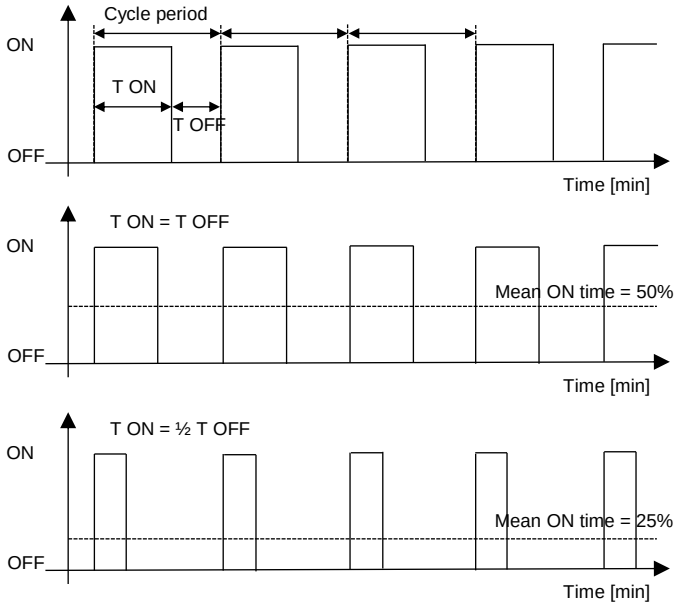


Integral Time is the amount of time necessary to repeat the value of the control variable of a purely proportional controller, when error is constant. For example, with a purely proportional controller with Proportional Band = 4 K, if Setpoint = 20°C and Measured Temperature = 18°C, the control variable will be 50%. If Integral Time = 60 minutes, if error remains constant, the control variable will be 100% after 1 hour, i.e. the controller will add to the control variable a contribution equal to the value due to its proportional part.

In heating and air conditioning systems, a purely proportional controller cannot guarantee reaching the Setpoint. An integral action is mandatory in order to reach the Setpoint: for this reason, the integral action is also called automatic reset.

6.7.3.3 PWM Proportional-Integral control

The proportional-integral PWM (Pulse Width Modulator) controller uses an analogue control variable to modulate the duration of the time intervals in which a binary output is in the On or Off state. The controller operates in a periodic manner over a cycle, and in each period it maintains the output to the On value for a time proportional to the value of the control variable. As shown in the figure, by varying the ratio between the ON time and the OFF time, the average time of activation of the output varies, and consequently the average intake of heating or cooling power supplied to the environment.



This type of controller is well suited for use with On / Off type actuators, such as relays and actuators for zone valves, which are less expensive (both for electrical and mechanical components) than proportional actuators. A distinctive advantage of this type of controller, compared with the raw On / Off controller already described, is that it eliminates the inertia characteristics of the system: it allows significant energy savings, because you avoid unnecessary interventions on the system introduced by the 2-point control with hysteresis and it only provides the power required to compensate for losses in the building.

Every time the user or the supervisor changes the desired temperature setpoint, the cycle time is interrupted, the control output is reprocessed and the PWM restarts with a new cycle: this allows the system to reach its steady state more quickly.

Terminal type	Proportional Band [K]	Integral Time [min]	Cycle Period [min]
Radiators	5	150	15-20
Electrical heaters	4	100	15-20
Fan-coil	4	90	15-20
Floor radiant panels	5	240	15-20

Guidelines for choosing the proper parameters of a PMW Proportional-Integral controller:

- Cycle time: for low-inertial systems such as heating and air conditioning systems, short cycle times must be chosen (10-15 minutes) to avoid oscillations of the room temperature.
- Narrow proportional band: wide and continuous oscillations of the room temperature, short setpoint settling time.
- Wide proportional band: small or no oscillations of the room temperature, long setpoint settling time.
- Short integral time: short setpoint settling time, continuous oscillations of the room temperature.
- Long integral time: long setpoint settling time, no oscillations of the room temperature.

6.7.4 Setpoint management

The pushbutton is not equipped with any local interface to control the integrated room thermostat, therefore the temperature setpoint modifications need to be managed through communication objects coming from a supervisory device.

Five setpoint management modes are foreseen:

- Single setpoint
- Relative setpoints, heating/cooling switch over from bus
- Relative setpoints, automatic heating/cooling switch over
- Absolute setpoints, heating/cooling switch over from bus
- Absolute setpoints, automatic heating/cooling switch over

Single setpoint mode

In this mode, a unique communication object is exposed (*Input Setpoint*) to modify the desired temperature. This object can be updated cyclically or on event of change by the supervisory device. If power goes down, the last value is retained into the pushbutton's non-volatile memory. In case the object is not updated, the temperature controller acts anyway on default setpoints (both heating and cooling) set in the application program during commissioning.

If a temperature controller is set on both heating and cooling mode, it is necessary that the supervisory device also updates the input seasonal mode object (*Heating/cooling status in*, [1.100] DPT_Heat_Cool) in order to coherently switch over the controller's action.

If window contacts for energy saving are used, when detecting an open window the input setpoint freezes and the pre-set building protection setpoint is activated (the relative communication object is exposed and is different in heating or cooling mode).

Relative setpoints, heating/cooling switch over from bus

In this mode, 4 communication objects are exposed, one for each operating mode:

- Comfort setpoint
- Stand-by offset
- Economy offset
- Building protection setpoint

Stand-by and economy setpoints are represented as attenuations to the comfort setpoint in order to facilitate the supervisor management: by uniquely modifying the comfort setpoint, references for attenuated modes are automatically transferred. The values modified from bus are retained in the pushbutton's non-volatile memory.

With this mode, the supervisory device can develop an hour-based time scheduling by sending to the pushbutton the current operating mode (comm. obj. *HVAC mode in* [20.102] DPT_HVACMode). The default value for *HVAC mode in* corresponds to the comfort setpoint value.

Same as single setpoint management, if the temperature controller is set as both heating and cooling mode, it is necessary that the supervisory device also updates the input seasonal mode object (*Heating/cooling status in*, [1.100] DPT_Heat_Cool) in order to coherently switch over the controller's action.

Relative setpoints, automatic heating/cooling switch over

In this mode, 3 communication objects are exposed, for all operating modes:

- Comfort heating setpoint
- Building protection heating setpoint
- Building protection cooling setpoint

Stand-by and economy setpoints are represented as attenuations to the comfort setpoint and can only be modified in the application program during commissioning: by uniquely modifying the comfort setpoint, references for attenuated modes and for comfort cooling setpoint mode (through switch over dead band) are automatically transferred. The values modified from bus are retained in the pushbutton's non-volatile memory.

With this mode, the supervisory device can develop an hour-based time scheduling by sending to the pushbutton the current operating mode (comm. obj. *HVAC mode in* [20.102] DPT_HVACMode). The default value for *HVAC mode in* corresponds to the comfort setpoint value.

The switch over between operating modes is automatic and the information can be sent to other devices through communication object *Heating/cooling status out*, [1.100] DPT_Heat_Cool). Please refer to the section about heating/cooling switch over to learn more about switch over modes.

6.7.5 Operating modes

In Single Setpoint mode, 2 levels for each operating mode are available:

- Temperature setpoint
- Building protection setpoint

Time scheduling for attenuation can be realized by the supervisor, by directly modifying the temperature setpoint.

In Relative Setpoint mode, 4 different operating modes are available, which are mutually exclusive to one another:

- comfort;
- stand-by;
- economy;
- building protection.

Through ETS application program, it is possible to assign 2 different setpoint values to each operating mode, for comfort and building protection level, and two different attenuation levels for stand-by and economy,

corresponding to both heating and cooling. Stand-by and economy setpoints are represented as attenuations to the comfort setpoint in order to facilitate the supervisor management: by uniquely modifying the comfort setpoint, references for attenuated modes are automatically transferred.

Each setpoint, except when automatic heating/cooling switch over is active, is exposed through communication objects. Setpoints and attenuations can be modified remotely through the exposed communication objects. The building protection setpoint intervention must be planned in ETS application program, as these parameters concern the safety and protection of the plant's components (especially during heating).

6.7.6 Heating/cooling switch over

The switch over between both heating and cooling mode can take place in 2 ways:

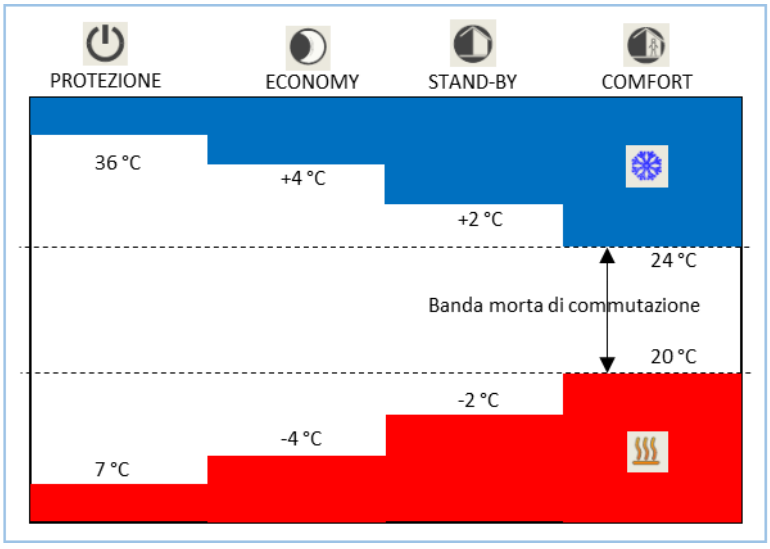
1. from KNX bus, through a communication object;
2. automatically, through a command from the internal logic of the device;

Switchover from bus

In mode 1, the switch over command is issued through KNX bus and therefore it is performed by a different KNX device, e.g. the ekinex® Touch&See unit. The integrated temperature controller acts as a "slave": the switch over is carried out by input communication object [DPT 1.100 heat/cool].

Automatic switch over

Mode 2 is suitable for applications with heating / cooling systems with a 4-pipe configuration (e.g. fan-coils or radiant ceiling panels). Also in this case the information can be transmitted on the bus through an output communication object [DPT 1.100 heat/cool]; the difference with mode 1 is that the switch over is performed automatically by the machine, basing on the values of current temperature and setpoint. The automatic switch over is achieved by introducing a dead band as shown in the following figure.



The figure shows that, as long as the actual measured temperature is below the heating mode setpoint, the heating mode is selected; similarly, if the value is greater than the cooling setpoint, then cooling mode is

selected. If the value is within the dead band, the operation mode remains unchanged until the value itself passes over the threshold value associated with the opposite mode.

The 4 setpoints for heating mode and the 4 setpoints for cooling mode are not exposed through communication objects to avoid inconsistencies between the different levels of temperature. In this case, a single communication object is published, which corresponds to the comfort heating setpoint. Every time this parameter is changed, the whole dead band changes with it, as well as all setpoints related to the 4 operating modes: the automatic switch over is then triggered outside the defined dead band.

6.7.7 Window switch management

Window switch management is an optional feature, oriented to energy saving, which becomes available only if the *Temperature control* function is enabled in the application program.

Whenever a condition of opened window is detected, the operating mode is forced to “building protection” and it remains forced as long as the open window condition is active. The program provides a time delay parameter for detection, in order to discriminate between an occasional short-term opening (e.g. to provide air exchange in the room) from an unintentional opening that justifies the power-saving function to be recalled.

The operating mode determined from Window switch management has priority on all operating mode settings imposed by the scheduler (in case *Setpoint management* = relative setpoint)

The physical detection of window openings is normally performed through switches that can be connected to KNX input devices; the pushbutton exposes up to 2 1-bit communication objects (*Temperature control* tab ⇒ External sensors) which can be synchronized to the switches’ states.

The internal logic performs a logical OR operation of the acquired contacts: the energy saving function is therefore activated if at least one window switch activation is detected. In order to determine the physical state of the contact corresponding to the “open window” state, two different options can be selected:

- NC (normally closed): open contact stands for closed window, closed contact stands for open window;
- NO (normally open): open contact stands for open window, closed contact stands for closed window;

6.7.8 Valve protection function

The function is suitable for both heating and cooling systems that use water as thermal conveying fluid and are provided with motorized valves for the interception of a zone or of a single room. Long periods of inactivity of the system can lead to the blockage of valves: to prevent this, the room temperature controller may periodically send a command to open / close the valve in the period of inactivity of the system. This possibility is further defined by the frequency and duration of the valve control.

6.7.9 Temperature control alarm

The integrated temperature controller can stop the internal control algorithm for one of the following reasons:

- For an external event, which can be configured and linked to the *Thermal generator lock* communication object;
- For an internal temperature sensor's fault (measured room temperature too low while NTC resistance value is too high or vice versa);
- For a timeout (data not updated by the bus) when a weighted mean between the internal sensor's value and an auxiliary external sensor's value is used.

When one of these events occur, the internal controller stops the control algorithm and the command output is taken to complete closing position (OFF or 0%): this state is signalled through the communication object *Temperature control alarm*.

6.8 Logic functions

The KNX pushbutton allows to use some useful logic functions (AND, OR, NOT and exclusive OR) in order to implement complex functions in the building automation system.

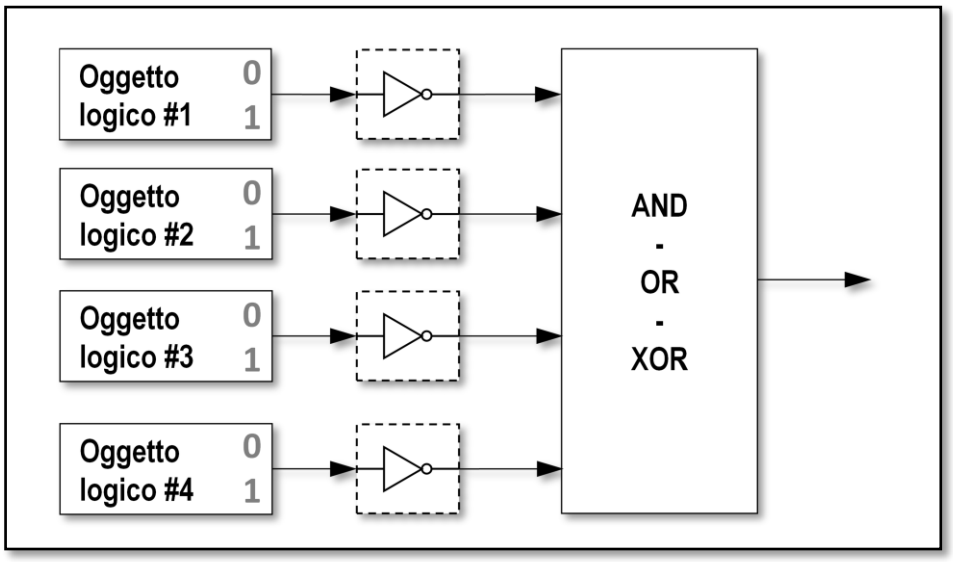
You can configure:

- 8 channels of logical functions
- 4 inputs for each channel

Each object value, if desired, can be individually inverted by inserting a NOT logic operator.

The inputs created by the objects are then logically combined as shown in the following figure:

Figure 6 – Logic combination function



The logic block on the right side of the figure has the following function, based on the selected operation:

- OR – the output is ON if at least one input is ON;
- AND – the output is ON if all inputs are ON;
- XOR – the output is ON if an odd number of inputs is ON;

This last function is more intuitive when there are only 2 inputs: in this case, the output is ON when one input or the other one is ON, but not the two of them simultaneously.

Please note that in this description, with "input" and "output" we refer only to the logic block; for the device operation, the effective "inputs" are given by communication objects, so also the possible activation of NOT logic operators has to be considered.

The following figures show the basic logic functions, assuming 2 inputs and only one logic communication object:

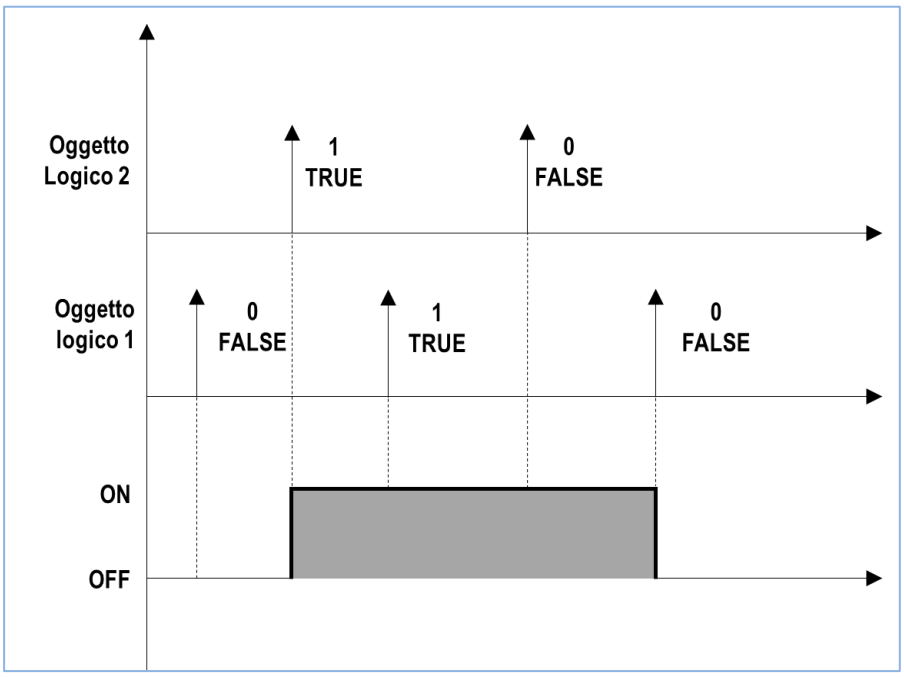


Figure 7 – Logic function OR

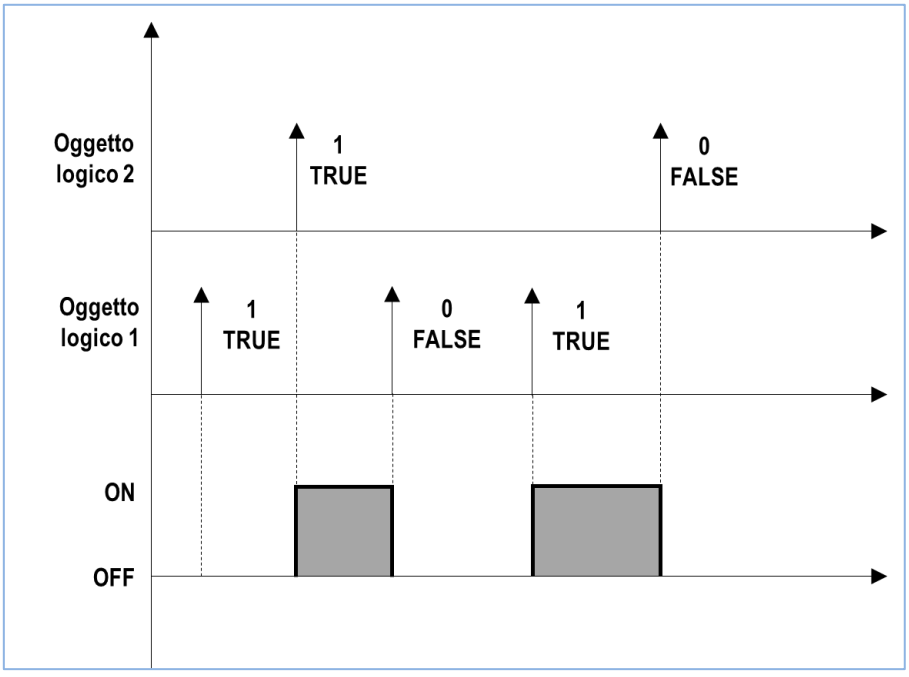


Figure 8 – Logic function AND

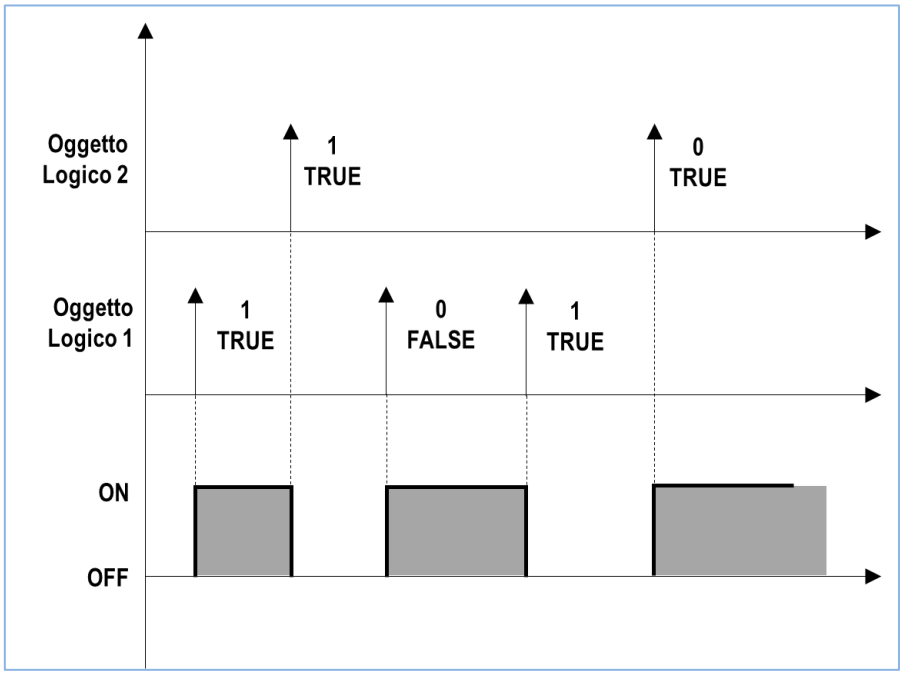


Figure 9 – Logic function Exclusive OR (XOR)

For each channel, a parameter *Delay after bus voltage recovery* is available: this parameter represents the time interval between the bus voltage recovery and the first reading of the input communication objects for evaluating the logic functions.

The communication function representing the logic function output is sent on the bus on event of change; alternatively, a cyclic sending can be set.

7 Application program for ETS

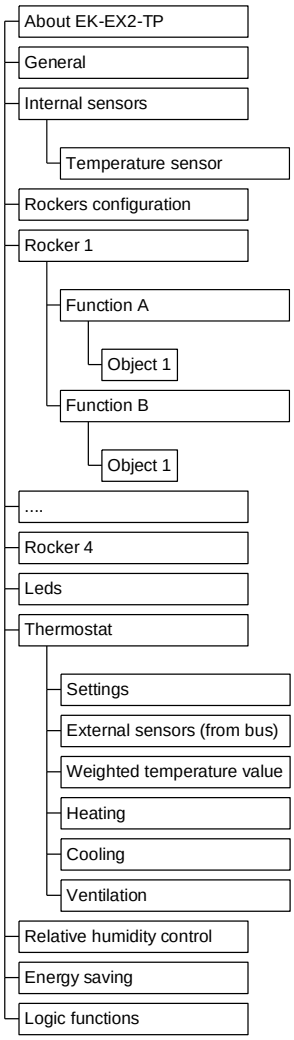
In the following chapters, there is the list of folder, parameters and communication objects of the application program.

Every channel, and every input or input pair under a channel, offers the same set of communication objects and parameters, but they may all be independently configured.

Hereafter, all channel-specific settings are listed grouped by channel; a generic channel number is referenced as “x” (where x = 1…4), while a generic input is referenced as “xx” (xx = 1A, 1B, 2A, ... 4B).

The parameter values highlighted in bold represent the default value.

The device settings are divided in two main groups: the *general* settings and the *channel-specific* settings. The settings are grouped in folders. The following figure shows the tree structure of the application program, with the main folders:



In order to use the device as a temperature sensor or as a room temperature controller it is sufficient to enable the temperature sensor in the *Internal sensors* folder. Consequently, also the *Temperature control* folder is activated: therefore, it is possible to select an auxiliary temperature sensor to perform a weighted mean with the main sensor and it is possible to configure the controller's options for room temperature.

7.1 About EK-ED2-TP and EK-E13-TP

The folder ***About EK-ED2-TP*** is for information purposes only and does not contain parameters to be set. The information given is:

Application software for ETS4
Version 4.00 (or later)
2-4 rockers pushbutton

Ekinex S.p.A.
Via Novara, 37
I-28010 Vaprio d'Agogna (NO) Italy
www.ekinex.com
info@ekinex.com

The folder ***About EK-E13-TP*** is for information purposes only and does not contain parameters to be set. The information given is:

Application software for ETS4
Version 4.00 (or later)
KNX pushbutton series 71

Ekinex S.p.A.
Via Novara, 37
I-28010 Vaprio d'Agogna (NO) Italy
www.ekinex.com
info@ekinex.com

7.2 General settings

The parameters in this section define the overall behaviour of the device, including the setting that defines which and how many channels are available.

Parameter name	*Conditions*	*Values*
Rockers configuration		*See Fig. 9 and 10 for available options*
	Specifies the configuration of installed rocker plates, thereby determining how physical pushbuttons will be associated to logical inputs and coupled in rocker pairs.	
Leds intensity from bus	-	yes / **no**
	Specifies whether the intensity value of LEDs should be set through a communication object.	
Leds intensity	Leds intensity from bus = no	0%..100% (**50%**)
	Fixed intensity value of LEDs.	
Delay after bus voltage recovery	-	hh:mm:ss.fff (**00:00:04.000**)
	Delay before bus telegrams can be sent after a recovery of the bus voltage. The delay time affects the transmission generated by an event as well as the cyclical transmission. For the cyclical transmission: after the delay time finished, the cycle restarts and the first telegram will be sent after the cycle time.	
Technical alarm	-	enabled / **disabled**
	Enables a communication objects that activates an alarm indication through a bus telegram. The indication is made by flashing the four LEDs at the corners of the device. *This indication is made available to the user for any purpose he sees fit (not necessarily an actual alarm).*	
Room temperature controller		enabled / **disabled**
	Enables the folder containing the parameters for room temperature control.	
Logic functions		enabled / **disabled**
	Enables the folders to configure AND, OR e XOR logic functions and their relative input and output communication objects.	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Technical alarm	Technical alarm = enabled	1 bit	C–W--	[1.005] alarm	1
Temperature value	Temperature sensor = enabled	2 Byte	CR–T-	[9.001] temperature (°C)	240

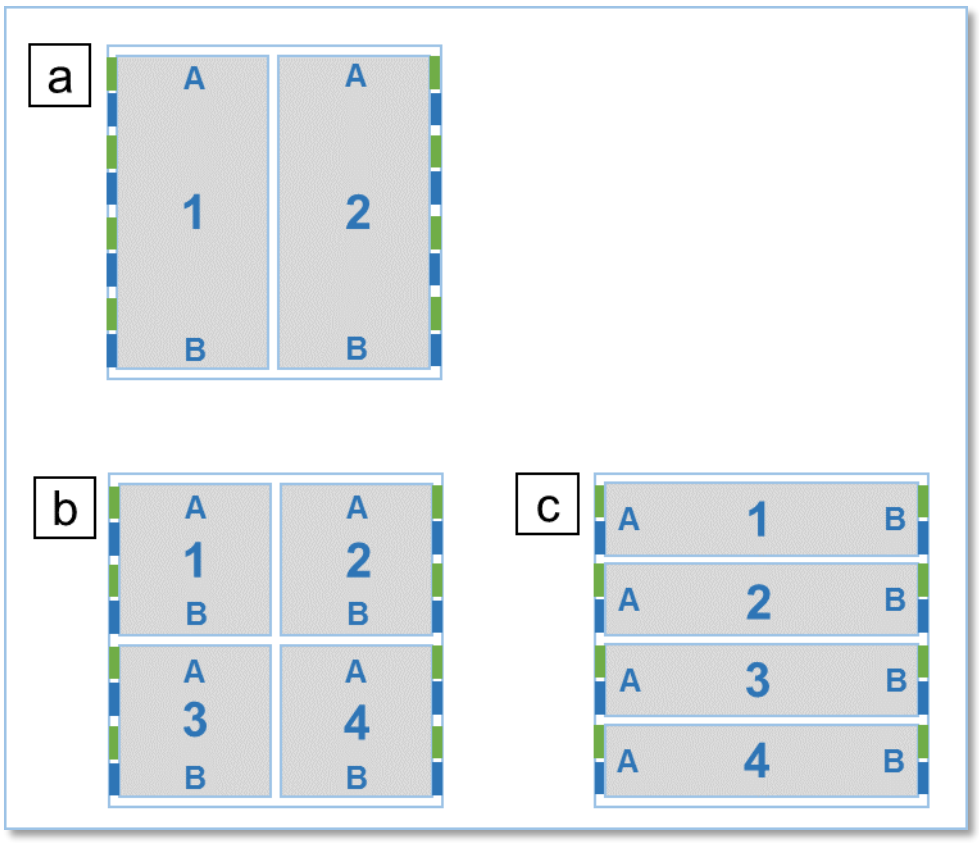


Figure 10A – 'BG version rockers' combination version for EK-ED2-TP

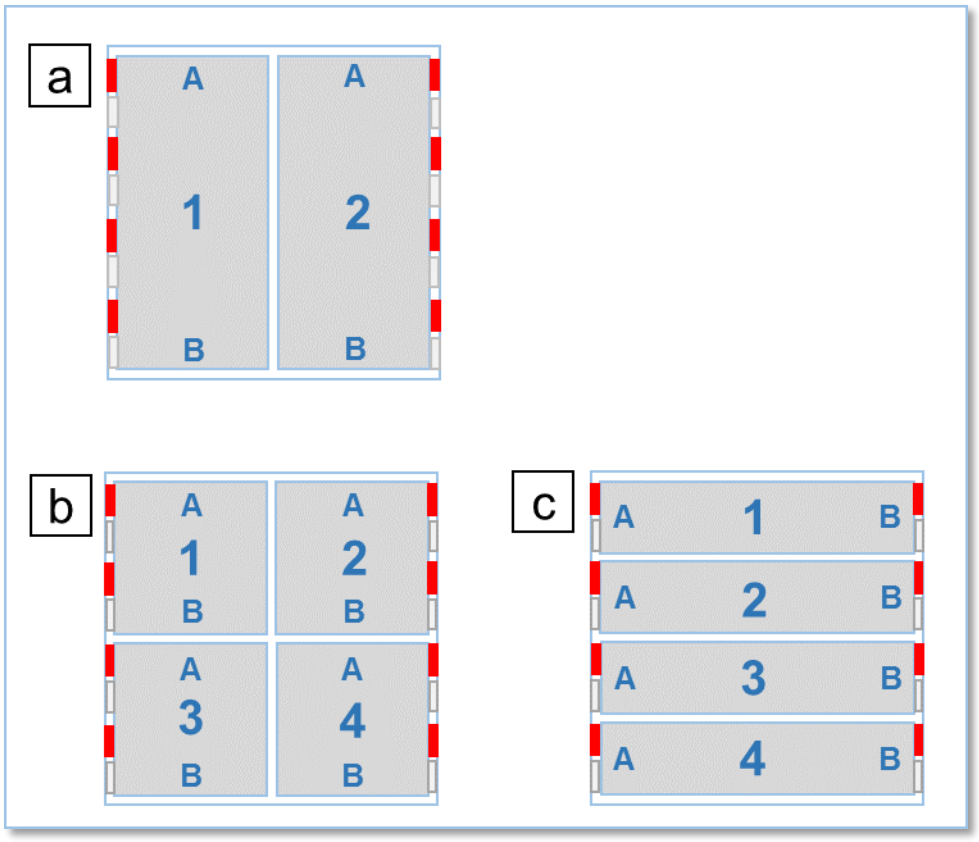


Figure 11A – 'RW' version rockers' combination version for EK-ED2-TP

The rockers' configuration is equivalent in BG (green and blue leds) and in RW (red and white).

Available rocker plate configuration options for 2-4 rocker pushbutton:

a. 2 vertical rectangular rockers
b. 4 square rockers
c. 4 rectangular horizontal rockers

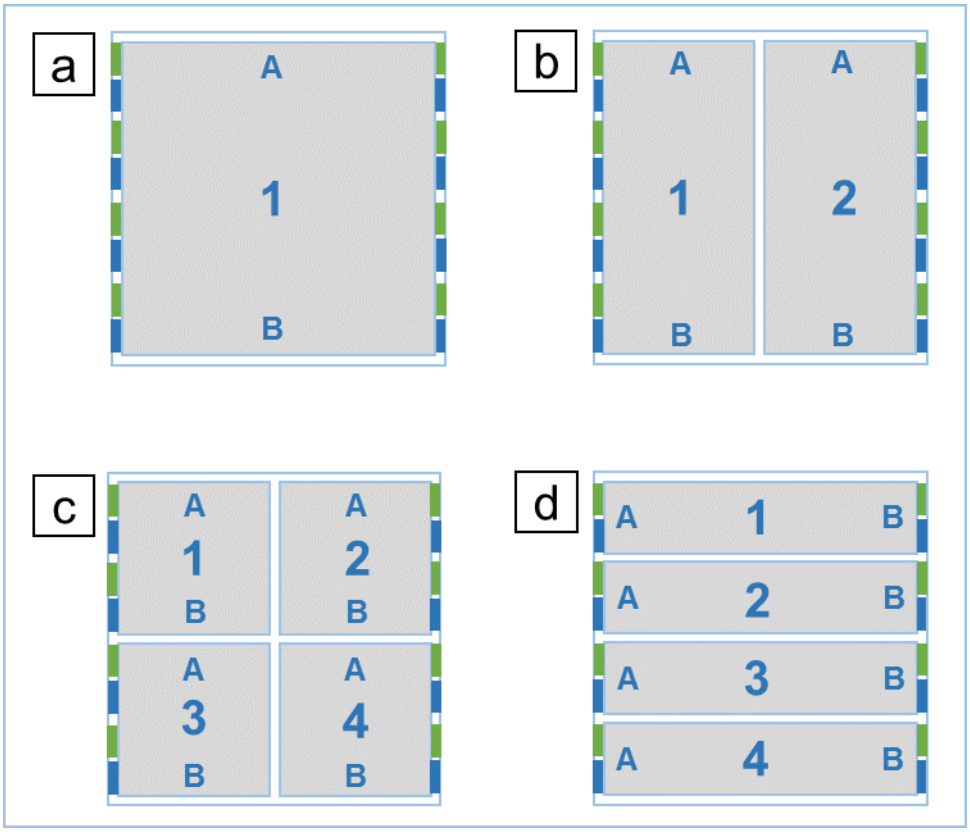


Figure 9B – 'BG version rockers' combination version for EK-E13-TP

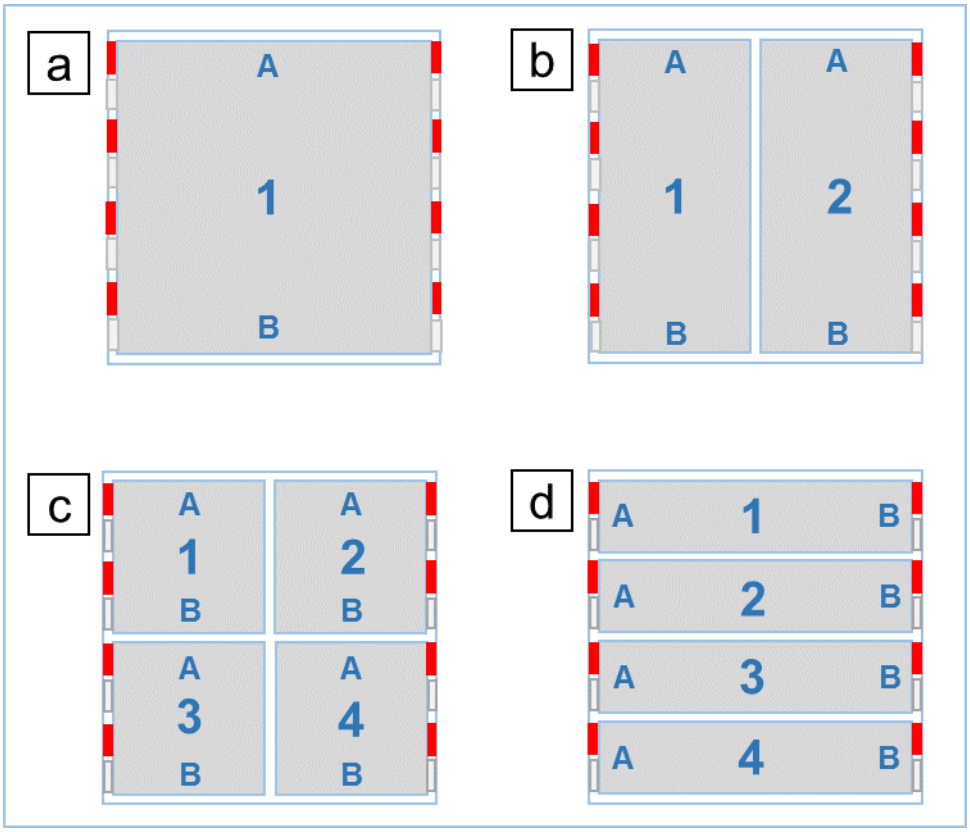


Figure 12B – 'RW version rockers' combination version for EK-E13-TP

The rockers' configuration is equivalent in BG (green and blue leds) and in RW (red and white).

Available rocker plate configuration options for 1-4 rocker pushbutton:

d. 1 single square rocker*
e. 2 vertical rectangular rockers
f. 4 square rockers
g. 4 rectangular horizontal rockers

7.3 Internal sensors

Parameter name	*Conditions*	*Values*
Temperature sensor		**disabled / enabled**
	Enables the temperature sensor by making the corresponding communication object available.	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Temperature value	Temperature sensor = enabled	2 Byte	CR–T-	[9.001] temperature (°C)	240

7.3.1 Temperature sensor

Parameter name	*Conditions*	*Values*
Filter type	Temperature sensor = enabled	low **average** high
	Low = average value every 4 measurements *Average = average value every 16 measurements* *High = average value every 64 measurements*	
Measured temperature correction	Temperature sensor = enabled	**0°C** [range -2,5°C … +2,5°C]
	Offset steps are 0.1°C in order to perform a more effective sensor calibration.	
Minimum change of value to send [K]	Temperature sensor = enabled	**0,5** [range 0 … 5]
	If the parameter is set to 0, no value is sent for change	
Cyclic sending interval	Temperature sensor = enabled	**no sending** [other values in range 30 s … 120 min]
Threshold 1	Temperature sensor = enabled	**not active** below above
Value [°C]	Temperature sensor = enabled, Threshold 1 = above or below	**7** [range 0 … 50]
Threshold value update from bus		**no** / yes
Threshold lock enable		**no** / yes
Behaviour at lock		**none** / off / on
Behaviour at bus recovery		**previous state** lock unlock
…		

Parameter name	*Conditions*	*Values*
Threshold 2	Temperature sensor = enabled	**not active** below above
Value [°C]	Temperature sensor = enabled, Threshold 2 = above or below	**45** [range 0 … 50]
Hysteresis	Temperature sensor = enabled, Threshold 1 e/o Threshold 2 = above or below	**0,4 K** [other values in range 0,2 K ... 3 K]
Cyclic sending interval	Temperature sensor = enabled, Threshold 1 e/o Threshold 2 = above or below	**no sending** [other values in range 30 s … 120 min]
Threshold value update from bus		**no** / yes
Threshold lock enable		**no** / yes
Behaviour at lock		**none** / off / on
Behaviour at bus recovery		**previous state** lock unlock

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Temperature threshold 1 - Switch	Temperature sensor = enabled, Threshold 1 = above or below	1 Bit	CR-T--	[1.001] switch	241
Temperature threshold 1 – Lock	Temperature sensor = enabled, Threshold lock enable = yes	1 Bit	C-W--	[1.1] DPT_Switch	242
Temperature threshold 1 – Value (from bus)	Temperature sensor = enabled, Threshold value update from bus = yes	2 Bytes	C-W--	[9.1] DPT_Value_Temp	243
Temperature threshold 2 - Switch	Temperature sensor = enabled, Threshold 2 = above or below	1 Bit	CR-T--	[1.001] switch	244
Temperature threshold 2 – Lock	Temperature sensor = enabled, Threshold lock enable = yes	1 Bit	C-W--	[1.1] DPT_Switch	245
Temperature threshold 2 – Value (from bus)	Temperature sensor = enabled, Threshold value update from bus = yes	2 Bytes	C-W--	[9.1] DPT_Value_Temp	246

7.4 Rockers configuration

Parameter name	*Conditions*	*Values*
Rocker x	-	disabled **independent or single** coupled copy parameters from rocker*
	Set operation mode for inputs corresponding to Rocker x. *The identification of which Rocker and associated input pushbuttons are corresponding to a given number (e.g. 1A – 2B – 4A etc.) is done according to parameter "General / Rockers configuration".* ** This option is only available for rockers no. 2 and above. If selected, the corresponding rocker can be made to perform the exact same kind of function as another specified rocker, but basing on different communication objects.* *This allows sparing time in configuring the device, at the same time assuring that there is no inconsistency between two rockers that are meant to be configured in exactly the same way.* *To assign the same configuration is just a shortcut for the selection of configuration options; it is in no way implied that the two rockers share any of the involved communication objects (each rocker has its own independent objects).*	
Function A	Rocker x = independent or single	**enabled /** disabled
	Enables or disables the capability to generate events for the first pushbutton of the rocker.	
Type	Rocker x = independent or single Function A = enabled	**send values or sequences** dimming shutter or venetian blind scene
	Determines the kind of function performed by the FIRST rocker input. *Further parameters for the selected function will appear in the individual rocker configuration sections (see below).*	
Function B	Rocker x = independent or single	disabled enabled in parallel with function A, as a single function **copy parameters from function A**
	Enables or disables the capability to generate events for the second pushbutton of the rocker. *If not disabled, the pushbutton can be given an own independent function (enabled),used as an "alias" of the first input (in parallel), or perform the exact same kind of function as first input (copy parameters), but possibly basing on a different communication object.*	
Type	Rocker x = independent or single Function B = enabled	**send values or sequences** dimming shutter or venetian blind scene
	Determines the kind of function performed by the SECOND rocker input. *Further parameters for the selected function will appear in the individual rocker configuration sections (see below).*	
Type	Rocker x = coupled	switch **dimming** shutter or venetian blind
	Determines the kind of function performed by the FIRST and SECOND rocker input. *Further parameters for the selected function will appear in the individual rocker configuration sections (see below).*	
Rocker to copy from	Rocker x = copy parameters from rocker (x > 1)	1..4*
	** The values that can be chosen obviously do not include the number of the rocker for which the selection is made.*	

7.4.1 Independent or single: send values or sequences

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Switching status [type], object n	Rocker x = independent or single Function x = enabled Type = send values or sequences	*according to the configuration* (**1-bit**)	CRWTU	*according to the configuration* (**[1.001] switch**)	6, 23 (1A, 1B) 40, 57 (2A, 2B) 74, 91 (3A, 3B) 108, 125 (4A, 4B)
	Up to 8 objects can be defined for binding with the same event. *The listed CO numbers are those referring to object nr.1; the COs for each subsequent object are following in sequence.* *To obtain the CO numbers for object number n, just add (n-1) to the listed numbers.* *E.g.: COs associated to input 3A (of Rocker 3) have numbers from 81 to 89. The number of CO no. 5 is therefore 81+(5-1) = 85.* ***The size and type of the individual objects can be configured as described in following sections**.*				

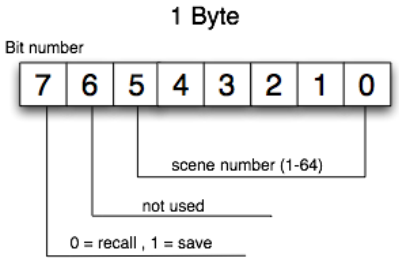
7.4.2 Independent or single: dimming

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Switching command	Rocker x = independent or single Function x = enabled Type = dimming	1 bit	CRWTU	[1.001] switch	14, 31 (1A, 1B) 48, 65 (2A, 2B) 82, 99 (3A, 3B) 116, 133 (4A, 4B)
	Send a command to a dimming actuator to switch the light on or off. *The command is triggered by a short press on the input.* *The value sent can be a fixed value or it can be toggled at each input activation.*				
Rocker x – Dimming up / down / stop command	Rocker x = independent or single Function x = enabled Type = dimming	4 bit	CR-T-	[3.*] 3-bit control	15, 32 (1A, 1B) 49, 66 (2A, 2B) 83, 100 (3A, 3B) 117, 134 (4A, 4B)
	Send a command to a dimming actuator to change dimming intensity (brighter or darker). *Three values are used which mean start increase, start decrease or stop the change.*  *Increase/decrease values are sent when a long press action occurs and stop value on press release.* *The value sent can be a fixed value or it can be toggled at each input activation.*				

7.4.3 Independent or single: shutter or venetian blind

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Dedicated stop command	Rocker x = independent or single Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.017] trigger	14, 31 (1A, 1B) 48, 65 (2A, 2B) 82, 99 (3A, 3B) 116, 133 (4A, 4B)
	Immediately stop any movement of the blind. The object is sent on a short press if the blind mode is disabled and at the end of a long press if the venetian blind mode is enabled.				
Rocker x – Stop – step up/down command	Rocker x = independent or single Function x = enabled Type = shutter or venetian blind Blind mode = enabled	1 bit	CR-T-	[1.007] step	17, 34 (1A, 1B) 51, 68 (2A, 2B) 85, 102 (3A, 3B) 119, 136 (4A, 4B)
	Move the blind to fully open or fully closed position. The object is sent at the end of a long press.				
Rocker x – Move up / down command	Rocker x = independent or single Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.008] up/down	18, 35 (1A, 1B) 52, 69 (2A, 2B) 86, 103 (3A, 3B) 120, 137 (4A, 4B)
	Increase or decrease the opening of the blind stepwise. *The object is sent on a short press.*				

7.4.4 Independent or single: scene

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Scene number	Rocker x = independent or single Function x = enabled Type = scene	1 Byte	CR-T-	[17.*] Scene number [18.*] Scene control	19, 36 (1A, 1B) 53, 70 (2A, 2B) 87, 104 (3A, 3B) 121, 138 (4A, 4B)
	Store or recall a scene. The lowest 6 bits in the byte form the code of the scene, while the highest bit is the operation code (store or recall).  				

7.4.5 Coupled: switch

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Switching command	Rocker x = coupled Function x = enabled Type = switch	1-bit	CRWTU	[1.001] switch	14, 48, 82, 116

7.4.6 Coupled: dimming

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Switching command	Rocker x = coupled Function x = enabled Type = dimming	1 bit	CRWTU	[1.001] switch	14, 48, 82, 116
	See notes for independent input.				
Rocker x – Dimming up / down / stop command	Rocker x = independent or single Function x = enabled Type = dimming	4 bit	CR-T-	[3.*] 3-bit control	15, 49, 83, 117
	See notes for independent input.				

7.4.7 Coupled: shutter or venetian blind

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker x – Dedicated stop command	Rocker x = coupled Function x = enabled Type = shutter or venetian blind Blind mode = disabled	1 bit	CRWTU	[1.017] trigger	14, 48, 82, 116
	See notes for independent input.				
Rocker x – Stop – step up/down command	Rocker x = coupled Function x = enabled Type = shutter or venetian blind Blind mode = enabled	1 bit	CR-T-	[1.007] step	17, 51, 85, 119
	See notes for independent input.				
Rocker x – Move up / down command	Rocker x = coupled Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.008] up/down	18, 52, 86, 120
	See notes for independent input.				

7.5 Rocker *x: Function A/B* configuration

7.5.1 Independent or single

For the *independent or* single channel setting, all parameters listed below are referred to either Function A or Function B (whichever are enabled).

In the following sections, it is implicitly understood that for the listed parameters to appear, the corresponding functions xA and/or xB must be enabled.

The entries assigned to “Object *n*” are repeated so many times as the number of configured objects according to the *Number of Communication Objects* parameter.

For all Type values:

Parameter name	***Conditions***	***Values***
Lock function	-	enabled / **disabled**
	Enables or disables the capability of locking the input through a remote command (telegram).	
Lock function – Invert lock device signal	Rocker x = independent or single Type = send values or sequences	**not inverted** / inverted
	Allows interpreting a “lock activate” telegram as unlock and vice-versa.	
Lock function – Lock after bus recovery	Rocker x = independent or single Type = send values or sequences	**no** / yes
	If active, after returning from a bus failure or power-off the device will retain the lock status it had before. Otherwise (in the default case), the device will restart in the non-locked condition.	

7.5.2 Independent or single: Lock function enabled

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker xx – Lock function	Rocker x = Independent or single Lock function = enabled	1 bit	C-W--	[1.003] enable	5, 22 (1A, 1B) 39, 56 (2A, 2B) 73, 90 (3A, 3B) 107, 124 (4A, 4B)

When the lock function is enabled, for each input or rocker the user can define an action to execute when a lock or unlock command is received.

Details are shown in the following sections; a resume of all options is shown in the table below.

Channel mode	*Input type*	*Behaviour at locking*	*Behaviour at unlocking*
independent /	send values or sequences	**none** as close or short press as open or long press	
/ coupled	switching	**none** off on toggle	**none** off on as previous
independent coupled	dimming		
independent /	scene	none send first scene send second scene	
independent coupled	shutter or venetian blind	none up down	

7.5.3 Independent or single: send values or sequences

Parameter name	*Conditions*	*Values*
Number of communication objects	Rocker x = independent or single Type = send values or sequences	1…8 (**1**)
	Number of communication objects configured in association with the button event.	
Lock function – Behaviour at locking	Rocker x = independent or single Type = send values or sequences	**none** as close or short press as open or long press
	Allows performing the operation associated to the specified event when a locking command is received. You can choose between operations linked to two possible closing (or short press, depending on the configuration) or opening (or long press) events.	
Lock function – Behaviour at unlocking	Rocker x = independent or single Type = send values or sequences	**none** as close or short press as open or long press
	Allows performing the operation associated to the specified event when an unlocking command is received. You can choose between operations linked to two possible closing (or short press, depending on the configuration) or opening (or long press) events.	
Event	Rocker x = independent or single Type = send values or sequences	**close / open contact** short / long press
	Type of event that should be used as trigger for an action.	
Long press time	Rocker x = independent or single Type = send values or sequences Event = short / long press	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
Object n – Send delay	Rocker x = independent or single Type = send values or sequences	hh:mm:ss.ff (**00:00:00.00**)
	Delay before the object is transmitted on the bus. *By defining a delay after the event occurs and before the object value is sent, it is possible to associate a time defined sequence of values to an input event.*	
Object n – Send cyclically	Rocker x = independent or single Type = send values or sequences Number of communication objects = 1	none **off / value 1** on / value 2 both off and on / both values
	Defines which of the values, if any, must be cyclically retransmitted whenever activated. *The cyclical transmission is only available if the number of communication objects to link is 1.*	
Object n – Cyclic sending interval	Rocker x = independent or single Type = send values or sequences Number of communication objects = 1 Send cyclically ≠ none	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Object n – send dimension	Rocker x = independent or single Type = send values or sequences	**1 bit value** 2 bits value 1 byte unsigned value 1 byte percentage 1 byte signed value 2 bytes unsigned value 2 bytes signed value 2 bytes floating value
	Defines size and type of the values to be sent when an event occurs.	
Object n – Close or Short press	Rocker x = independent or single Type = send values or sequences send dimension = 1 bit	none on off **toggle**

Parameter name	*Conditions*	*Values*
	Rocker x = independent or single Type = send values or sequences send dimension = 2 bit	none disable enable off / up **enable on / down** enable off / up ↔ disable enable on / down ↔ disable enable off / up ↔ enable on / down
	Rocker x = independent or single Type = send values or sequences send dimension = *any byte value*	none **send value 1** send value 2 send value 1 ↔ send value 2
	Value change behaviour caused by either a Close or a Short Press event (according to event configuration)	
Object n – Open or Long press	Rocker x = independent or single Type = send values or sequences send dimension = 1 bit	**none** on off toggle
	Rocker x = independent or single Type = send values or sequences send dimension = 2 bit	none **disable** enable off / up enable on / down enable off / up ↔ disable enable on / down ↔ disable enable off / up ↔ enable on / down
	Rocker x = independent or single Type = send values or sequences send dimension = *any byte value*	**none** send value 1 send value 2 send value 1 ↔ send value 2
	Value change behaviour caused by either an Open or a Long Press event (according to event configuration)	
Object n – Value 1	Rocker x = independent or single Type = send values or sequences send dimension = *any byte value*	0...255 (1 byte unsigned value) 0…100 (1 byte percentage) -128...127 (1 byte signed value) 0...65535 (2 bytes unsigned value) -32768... 32767 (2 bytes signed value) -671088...670597 (2 bytes floating value)
	First value available for association in send events	
Object n – Value 2	Rocker x = independent or single Type = send values or sequences send dimension = *any byte value*	*same as value 1*
	Second value available for association in send events	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker xx – Switching status [type] Object n	Rocker x = Independent or single Type = send values or sequences	*See table below*	CR-TU	*See table below*	6, 23 (1A, 1B) 40, 57 (2A, 2B) 74, 91 (3A, 3B) 108, 125 (4A, 4B)
	The listed CO numbers are those referring to object nr.1; the COs for each subsequent object are following in sequence. *To obtain the CO numbers for object number n, just add (n-1) to the listed numbers.* *E.g.: COs associated to input 3A (of Rocker 3) have numbers from 81 to 89. The number of CO no. 5 is therefore 81+(5-1) = 85.*				

Sizes and DPTs are as follows:

Size	*DPT*
1 bit	[1.001] switch
2 bit	[2.*] 1-bit controlled
1 byte unsigned value	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte percentage	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte signed value	[6.*] 8-bit signed value
2 bytes unsigned value	[7.*] 2-byte unsigned value
2 bytes signed value	[8.*] 2-byte signed value
2 bytes floating value	[9.*] 2-byte float value

7.5.4 Independent or single: dimming

Parameter name	*Conditions*	*Values*
Long press time	Rocker x = independent or single Type = dimming	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
Toggle mode	Rocker x = independent or single Type = dimming	**enabled /** disabled
	When enabled, causes the short press to toggle the on-off status of the destination CO; otherwise, a fixed status can be assigned to the short press.	
Long action	Rocker x = independent or single Type = dimming Toggle mode = enabled	**darker** brighter darker ↔ brighter
	Defines the function to be assigned to the long press. If the toggle mode is enabled, the Short press action is already defined as toggle.	
Short / Long action	Rocker x = independent or single Type = dimming Toggle mode = disabled	**off / darker** on / brighter off / darker ↔ brighter on / darker ↔ brighter
	Defines the function to be assigned to the long and short press.	
Send cyclically	Rocker x = independent or single Type = dimming	**none** off / value 1 on / value 2 both off and on / both values
	Defines which of the values, if any, must be cyclically retransmitted whenever activated.	
Cyclic sending interval	Rocker x = independent or single Type = dimming Send cyclically ≠ none	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Rocker x = independent or single Type = dimming	**none** off on toggle
	Value to be assigned to the object when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = independent or single Type = dimming	**none** off on as previous
	Value to be assigned to the object when an unlocking command is received.	

7.5.5 Independent or single: shutter or venetian blind

Parameter name	*Conditions*	*Values*
Long press time	Rocker x = independent or single Type = shutter or venetian blind	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
Toggle mode	Rocker x = independent or single Type = shutter or venetian blind	**enabled /** disabled
	When enabled, causes the short press to toggle the on-off status of the destination CO; otherwise, a fixed status can be assigned to the short press.	
Up / Down action	Rocker x = independent or single Type = shutter or venetian blind Toggle mode = disabled	**up** down
	Defines the movement direction to be assigned to the button press.	
Blind mode	Rocker x = independent or single Type = shutter or venetian blind	enabled / **disabled**
	If blinds mode is enabled, the device sends "full movement" telegrams on long press and "step" telegrams on short press; if it is disabled, the device sends "full movement" telegrams on long press and "stop" telegrams on short press.	
Lock function – Behaviour at locking	Rocker x = independent or single Type = shutter or venetian blind	**none** up down
	Operation to perform when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = independent or single Type = shutter or venetian blind	**none** up down
	Operation to perform when an unlocking command is received.	

7.5.6 Independent or single: scene

Parameter name	*Conditions*	*Values*
First scene number	Rocker x = independent or single Type = scene	1…64 (**1**)
	Main scene number to be assigned to button press. It is named "first" for the case that an alternative scene number is used.	
Learning mode	Rocker x = independent or single Type = scene	enabled / **disabled**
	When enabled, a long key press can be used to program the selected scene by storing the current parameters.	
Long press time	Rocker x = independent or single Type = scene Learning mode = enabled	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
Scene activation	Rocker x = independent or single Type = scene Learning mode = disabled	**send first scene only** toggle between two scenes
	Allows the key to be used to alternate between two different scenes.	
Second scene number	Rocker x = independent or single Type = scene Learning mode = disabled Scene activation = toggle between two scenes	1…64 (**2**)
	Alternate scene number to be assigned to button press.	
Lock function – Behaviour at locking	Rocker x = independent or single Type = scene	**none** send first scene send second scene
	Operation to perform when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = independent or single Type = scene	**none** send first scene send second scene
	Operation to perform when an unlocking command is received.	

7.5.7 Coupled

For a *coupled* channel, all the parameters are referred to the single menu entry for Function xA and xB.

In the following sections, it is implicitly understood that for the listed parameters to appear, the corresponding functions xA and xB must be enabled.

For all Type values:

Parameter name	*Conditions*	*Values*
Lock function	Rocker x = coupled	enabled / **disabled**
	Enables or disables the capability of locking the input through a remote command (telegram).	

7.5.8 Coupled: Lock function enabled

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Rocker xx – Lock function	Rocker x = coupled Lock function = enabled	1 bit	C-W--	[1.003] enable	5, 39, 73, 107

7.5.9 Coupled: switch

Parameter name	*Conditions*	*Values*
xA and xB use	Rocker x = coupled Type = switch	**A on, B off** A off, B on
	Allows to invert side A and side B functionalities	
Send cyclically	Rocker x = coupled Type = switch	none **off / value 1** on / value 2 both off and on / both values
	Defines which of the values, if any, must be cyclically retransmitted whenever activated.	
Cyclic sending interval	Rocker x = coupled Type = switch Send cyclically ≠ none	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Rocker x = coupled Type = switch	**none** on off toggle
	Value to be assigned to the object when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = coupled Type = switch	**none** on off as previous
	Value to be assigned to the object when an unlocking command is received.	

7.5.10 Coupled: dimming

Parameter name	*Conditions*	*Values*
Long press time	Rocker x = coupled Type = dimming	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
xA and xB use	Rocker x = coupled Type = dimming	**A increases, B decreases** A decreases, B increases
Send cyclically	Rocker x = coupled Type = dimming	none **off / value 1** on / value 2 both off and on / both values
	Defines which of the values, if any, must be cyclically retransmitted whenever activated.	
Cyclic sending interval	Rocker x = coupled Type = dimming Send cyclically ≠ no	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Rocker x = coupled Type = dimming	**none** on off toggle
	Value to be assigned to the object when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = coupled Type = dimming	**none** on off as previous
	Value to be assigned to the object when an unlocking command is received.	

7.5.11 Coupled: shutter or venetian blind

Parameter name	*Conditions*	*Values*
Long press time	Rocker x = coupled Type = shutter or venetian blind	hh:mm:ss.fff (**00:00:03.000**)
	Minimum push time for a press in order to be recognized as a long press.	
xA and xB use	Rocker x = coupled Type = shutter or venetian blind	**A up, B down** A down, B up
Blind mode	Rocker x = coupled Type = shutter or venetian blind	enabled / **disabled**
	If blinds mode is enabled, the device sends "full movement" telegrams on long press and "step" telegrams on short press; if it is disabled, the device sends "full movement" telegrams on long press and "stop" telegrams on short press.	
Lock function – Behaviour at locking	Rocker x = coupled Type = shutter or venetian blind	**none** up down
	Operation to perform when a locking command is received.	
Lock function – Behaviour at unlocking	Rocker x = coupled Type = shutter or venetian blind	**none** up down
	Operation to perform when an unlocking command is received.	

For other communication objects related to *coupled* mode, please refer to the general *Rockers Configuration* section.

7.6 LED configuration

Following parameters are repeated for each of the available LEDs.

LED parameters settings are always listed grouped by rocker (regardless whether the inputs are coupled or not): for each rocker *x*, available LEDs are marked as LED First Colour (green in BG version, red in RW) and LED Second Colour (blue in BG version, white in RW).

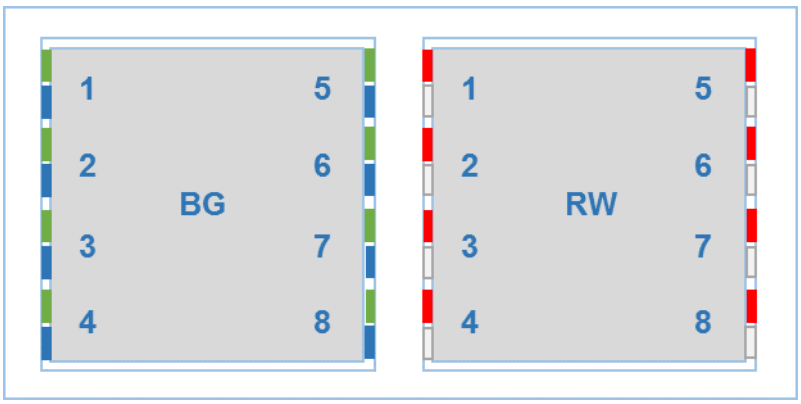
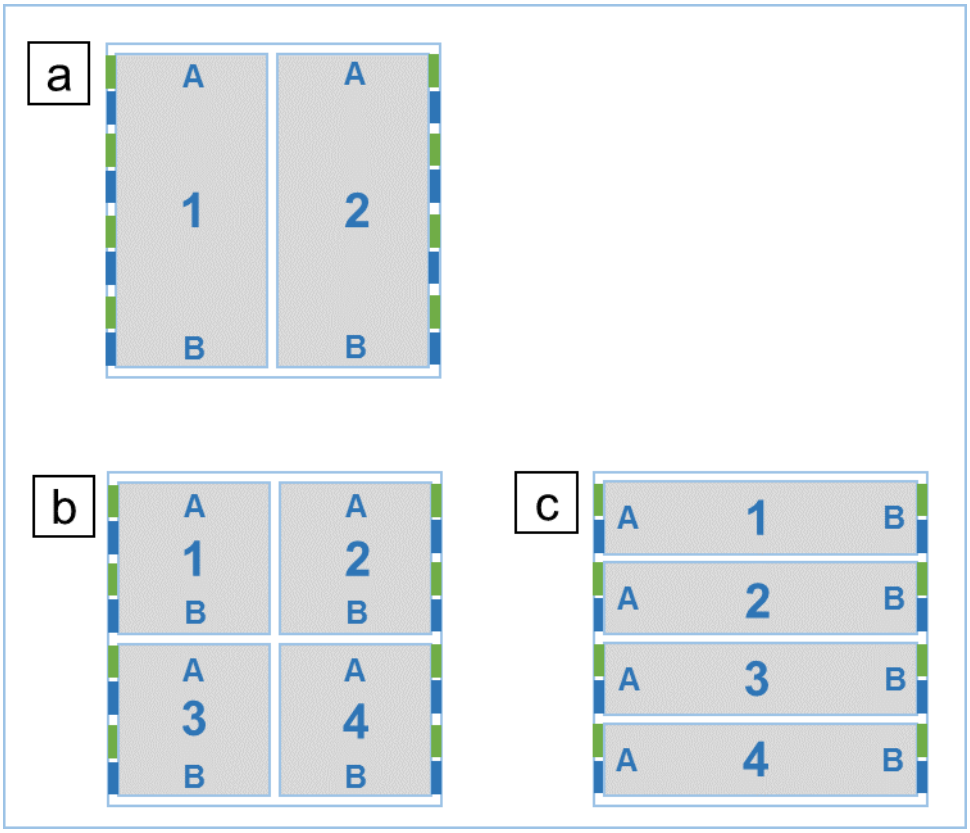


Figure 13A - Number convention for LED couples EK-ED2-TP and EK-E13-TP

Parameter name	***Conditions***	***Values***
LED X First colour/Second colour	-	fixed **when contact closed** status from bus
Always	LED X First colour/Second colour = fixed	on / **off**
	Status for a fixed LED condition	
Off delay	LED X First colour/Second colour = when contact closed	hh:mm:ss.ff (**00:02:00.00**)
	Delay before LED goes off	
Blinking	LED X First colour/Second colour = status from bus	**no** / yes
Signal from bus	LED X First colour/Second colour = status from bus	**not inverted** / inverted
	Specifies whether the LED status from the bus should be inverted, i.e. LED on when an "off" command is received on the communication object. *This feature is useful because the condition for the LED to switch on can be linked to a communication object related to the state of another device, which may have an opposite logic.*	

Object name	***Conditions***	***Size***	***Flags***	***DPT***	***No. Comm. Obj.***
LED X – First colour	LED first colour 1 = status from bus	1 bit	CRWTU-	[1.001] switch	141, 143, 245, 147, 149, 151, 153, 155
LED X - Second colour	LED second colour 1 = status from bus	1 bit	CRWTU-	[1.001] switch	142, 144, 146, 148, 150, 152, 154, 156



		Rocker combinations			
	LED	a)	b)	c)	d)
Independent A/B functions	1	1A	1A	1A	1A
Parallel A/B functions		1A	1A	1A	not allowed
Independent A/B functions	2	1A	1A	1B	2A
Parallel A/B functions		1A	1A	1A	not allowed
Independent A/B functions	3	1B	1B	3A	3A
Parallel A/B functions		1A	1A	3A	not allowed
Independent A/B functions	4	1B	1B	3B	4A
Parallel A/B functions		1A	1A	3A	not allowed
Independent A/B functions	5	1A	2A	2A	1B
Parallel A/B functions		1A	2A	2A	not allowed
Independent A/B functions	6	1A	2A	2B	2B
Parallel A/B functions		1A	2A	2A	not allowed
Independent A/B functions	7	1B	2B	4A	3B
Parallel A/B functions		1A	2A	4A	not allowed
Independent A/B functions	8	1B	2B	4B	4B
Parallel A/B functions		1A	2A	4A	not allowed

This table shows the LEDs (according to the adopted convention) activated with a press on the areas, according to the adopted rocker combination.

7.7 Temperature and Relative humidity control

The *Temperature control* folder allows the pushbutton to be configured as a room temperature controller and also allows filtrating, through a weighted mean average, the reading from the internal sensor with the reading from an auxiliary sensor added on the bus.

The folder is active only if the room temperature sensor has been enabled: *Internal sensors* ⇒ Temperature sensor = enabled.

It includes the following secondary folders:

- Settings
- Sensors from bus
- Weighted temperature value
- Heating
- Cooling

7.7.1 Settings

The *Settings* includes the parameters to perform the basic configuration of the room temperature controller:

- Heating and cooling function
- Setpoint management mode: single or relative
- Separated actuator command or common in heating/cooling
- Heating/cooling switch over type
- Valve protection function activation

This folder is always active.

Parameter name	*Conditions*	*Values*
Thermostat function		**disabled** heating cooling both heating and cooling
Setpoint management		Single setpoint **Absolute setpoints** Relative setpoints
	In case the option "Single setpoint" in selected and Thermostat function = heating, the temperature controller acts on heating mode; in case Thermostat function = cooling, the temperature controller acts on cooling mode. *In case Thermostat function = both heating and cooling, the current seasonal mode needs to be specified by the proper communication object.*	
Command Communication Object	Thermostat function = both heating and cooling	**separated** / unique
Heating–cooling switch over	Thermostat function = both heating and cooling, Setpoint management = Relative setpoints	**from bus** / automatic
	In case Setpoint management = single setpoint, the heating-cooling switch over must be carried out from bus.	

Parameter name	*Conditions*	*Values*
Cyclic sending interval setpoint		**no sending** [other values in range 30 s … 120 min]
	In case Setpoint management = single setpoint, the actual setpoint value takes only into account the actual state of the contacts window (if the corresponding function is enabled). *In case Setpoint management = relative setpoints, the actual setpoint value also depends on the operating mode set manually by the user or automatically by another KNX supervising device with the possibility of time scheduling.*	
End of manual operation	Setpoint management = Absolute setpoints or Relative setpoints	**till first telegram from bus** [other values in the range 30 min…48 h]
	It defines the end of manual operation modes.	
Valve protection function		**disabled / enabled**
	It enables the function that activates the drive for the valve control during periods of inactivity of the system.	
Frequency	Valve protection function = enabled	once a day, **once a week**, once a month
Time interval	Valve protection function = enabled	**10 s** [other values in range 5 s … 20 min]

All communication object are active only if parameter *Settings* ⇒ Thermostat function ≠ disabled

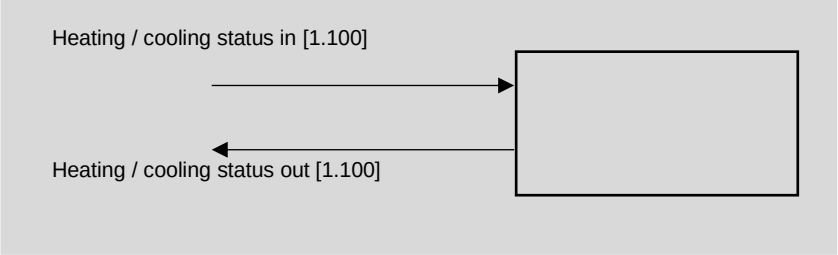
Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Thermal generator lock		1 Bit	-WC---	[1.001] switch	308
Thermostat – Room temperature control alarm		1 Bit	CR-T-	[1.005] alarm	296
Thermostat – Actual setpoint		2 Byte	CR-T--	[9.001] temperature (°C)	269
Thermostat – Heating/cooling status out	Always exposed	1 Bit	CR-T--	[1.100] heating/cooling	262
	The communication object is sent over the bus after an internally elaborated switching event. **[1.100] DPT Heat/Cool 1 Bit** **0 = CoolT** **1 = HeatingT**				
Thermostat – Heating/cooling status in	Thermostat function = both heating and cooling, Heating–cooling switch over = from bus	1 Bit	C-W---	[1.100] heating/cooling	263
	The communication object is received from the bus. At the switching event the internal regulator switches the conduction mode.				

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat - HVAC mode in	Setpoint management = Absolute setpoints or Relative setpoints	1 Byte	C-W---	[20.102] HVAC mode	264
	Bits 5, 8 are reserved. **[20.102] DPT HVAC Mode 1 Byte** AUTO 0 0 0 0 COMFORT 0 0 0 1 STAND-BY 0 0 1 0 ECONOMY 0 0 1 1 PROTECTION 0 1 0 0				
Thermostat - HVAC forced mode in	Setpoint management = Absolute setpoints or Relative setpoints				265
Thermostat - HVAC manual mode	Setpoint management = Absolute setpoints or Relative setpoints	1 Byte	C-W---	[20.102] HVAC mode	267
Thermostat – Chrono active status	Setpoint management = Absolute setpoints or Relative setpoints	1 Bit	CRWTU-	[1.011] state	268
Thermostat - HVAC mode out	Setpoint management = Absolute setpoints or Relative setpoints	1 Byte	C-W---	[20.102] HVAC mode	266

7.7.2 Monitoring and remote control of the conduction mode

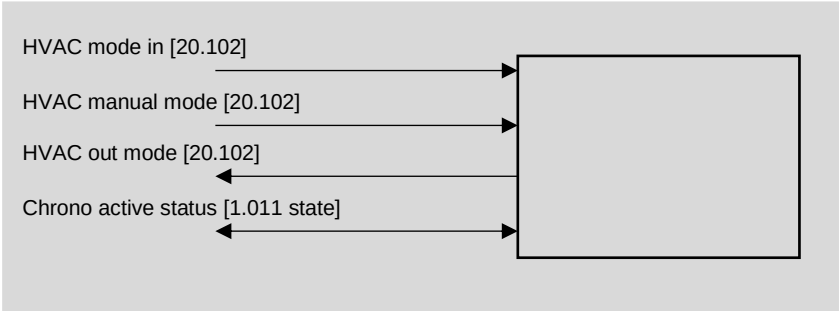
The communication objects indicated in the block diagram allows monitoring and modifying the current conduction mode forced on the temperature controller. The object *Heating/cooling status out* is always exposed, even when the thermostat function is set on heating or cooling only. When the function is set on both heating and cooling, the cyclic sending on bus can be enabled; anyway, the information about the actual conduction mode can be acquired with a reading request to this commiunication object.

The object *Heating/cooling status in* is exposed only when the function is both heating and cooling and the switching among the different modes is performed by the bus.



7.7.3 Remote operative mode modification

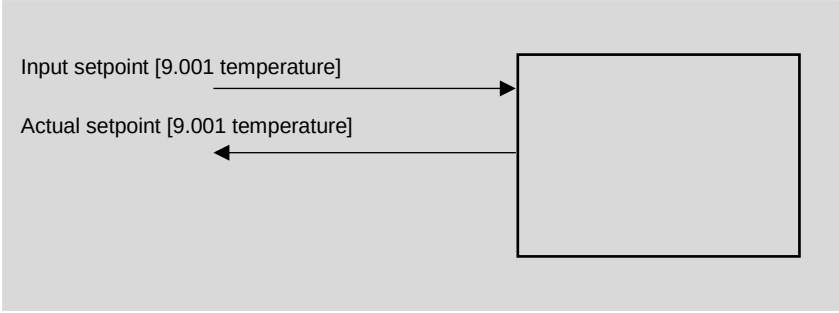
The communication objects shown in figure allow to monitor the operating mode (comfort, standy, economy and building protection) modifications performed locally by the user when interacting with the LCD display and the touch buttons of the room thermostat, or the operating mode forced by chrono program.

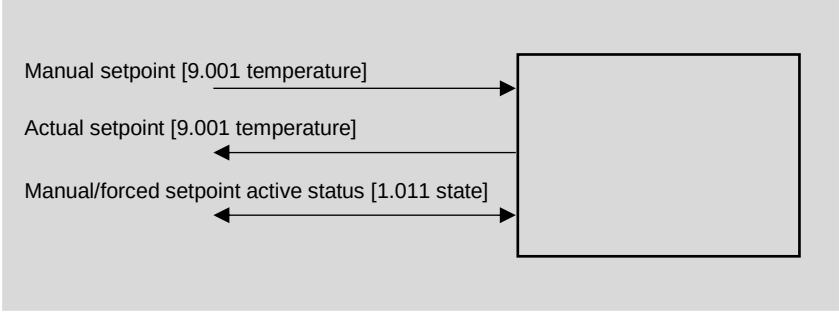


The C.O. *HVAC mode in* is associated to the chrono program. The C.O.s *HVAC mode out* and *HVAC chrono active status* allow the remote supervisor to discern the operating mode currently active on the room thermostat and also allow to understand if the chrono program is active or if attenuation is handled manually or not. The supervisor can set at any time a manual operating mode through C.O. *HVAC manual mode*; to start the chrono program remotely, the C.O. *HVAC manual mode* is to be set on value 0 = Automatic.

7.7.4 Remote Setpoint modification

These communication objects allow to perform the same modifications remotely, for example from a supervisor software.





Those objects are about the Setpoint forced modification: alternatively, the supervisor can act directly on the operating mode setpoints. The value of the C.O. *Actual setpoint* represents the current operative setpoint which the control algorithms are based on. The C.O. *Manual/forced setpoint active status* indicates (read request mode) if the forced mode is active (symbol M on the LCD display present). The supervisor can force at any time the actual setpoint by writing a new value directly into the C.O. *Manual setpoint*. The C.O. *Manual/forced setpoint active status* can also be used in writing to exit the active forced mode.

7.7.5 Sensors from bus

Sensors from bus are KNX devices (or traditional sensors interfaced to the bus by means of KNX devices) which send values or states to the pushbutton via bus. The folder allows activating an auxiliary temperature sensor and 2 window contacts to automatically recall the building protection setpoint: the automatic recall function must be individually configured for the 2 conduction modes in *Heating and/or Cooling* folder.

The internal pushbutton control system cyclically monitors the updating status of the values of the external sensors (from bus) when the timeout setting expires. In case no updated value has been received, the regulation function is suspended and the actuation valves are closed. An alarm is issued on the bus through communication object *Temperature control alarm* (please refer to *Settings* folder).

The folder is always active.

Parameter name	***Conditions***	***Values***
Room temperature (from bus)		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	
Cyclic reading each	Room temperature (from bus) = enabled	**no reading** other values in range [30s, …, 120 min]
Relative humidity	Relative humidity = enabled	**disabled** / enabled
Humidity CO dimension	Relative humidity = enabled	**1 byte** / 2 bytes
Cyclic reading each	Relative humidity = enabled	**no reading** other values in range [30s, …, 120 min]
Antistratification temperature		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	
Cyclic reading each	Antistratification temperature = enabled	**no reading** other values in range [30s, …, 120 min]
Outdoor temperature		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	
Cyclic reading each	Outdoor temperature = enabled	**no reading** other values in range [30s, …, 120 min]
Coil temperature		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	
Cyclic reading each	Coil temperature = enabled	**no reading** other values in range [30s, …, 120 min]
Floor temperature		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	

Parameter name	*Conditions*	*Values*
Cyclic reading each	Floor temperature = enabled	**no reading** other values in range [30s, …, 120 min]
Flow temperature		**disabled** / enabled
	Enables a temperature bus sensor. The measured value can be used to calculate a weighted mean average with the integrated temperature sensor.	
Cyclic reading each	Flow temperature = enabled	**no reading** other values in range [30s, …, 120 min]
Analog sensors timeout (from bus)		**00:05:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	The field has format hh:mm:ss (hours : minutes : seconds): the default value 00:05:00 corresponds to a timeout of 5 minutes. The value 00:00:00 means that the timeout of the analogic sensors is disabled.	
[…]		
Anticondensation		**disabled /** enabled
Signal	Anticondensation = enabled	**not inverted** / inverted
Cyclic reading each	Anticondensation = enabled	**no reading** other values in range [30s, …, 120 min]
Window contact 1		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Window contact 1 = enabled	**not inverted** / inverted
Cyclic reading each	Window contact 1 = enabled	**no reading** other values in range [30s, …, 120 min]
Window contact 2		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Window contact 1 = enabled	**not inverted** / inverted
Cyclic reading each	Window contact 1 = enabled	**no reading** other values in range [30s, …, 120 min]
Window contact 2		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Window contact 1 = enabled	**not inverted** / inverted
Cyclic reading each	Window contact 1 = enabled	**no reading** other values in range [30s, …, 120 min]
Presence sensor 1		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Presence sensor 1 = enabled	**not inverted** / inverted

Parameter name	*Conditions*	*Values*
Cyclic reading each	Presence sensor 1 = enabled	**no reading** other values in range [30s, …, 120 min]
Presence sensor 2		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Presence sensor 2 = enabled	**not inverted** / inverted
Cyclic reading each	Presence sensor 2= enabled	**no reading** other values in range [30s, …, 120 min]
Card holder contact		**disabled /** enabled
	I*t enables a bus sensor for detecting the state of opening / closing of a window or a door.*	
Signal	Card holder contact = enabled	**not inverted** / inverted
Cyclic reading each	Card holder contact = enabled	**no reading** other values in range [30s, …, 120 min]
Digital sensors timeout (from bus)		**00:05:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	The field has format hh:mm:ss (hours : minutes : seconds): the default value 00:05:00 corresponds to a timeout of 5 minutes. The value 00:00:00 means that the timeout of the digital sensors is disabled.	

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Room temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	247
Thermostat – Humidity (2 bytes, from bus)	enabled	2 Byte	C-WTU	[9.7] DPT_Value_Humidity	248
Thermostat – Humidity (1 byte, from bus)	enabled	2 Byte	C-WTU	[5.1] DPT_Scaling	249
Thermostat – Antistratification temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	250
Thermostat – Outdoor temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	251
Thermostat – Coil temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	252
Thermostat – Floor temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	253
Thermostat – Flow temperature (from bus)	enabled	2 Byte	C-WTU	[9.001] temperature (°C)	254

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Windows contact sensor 1 (from bus)	enabled	1 Bit	C-WTU	[1.019] window/door	255
Thermostat – Windows contact sensor 2 (from bus)	enabled	1 Bit	C-WTU	[1.019] window/door	256
Thermostat – Presence sensor 1 (from bus)	enabled	1 Bit	C-WTU	[1.18] DPT_Occupancy	257
Thermostat – Presence sensor 2 (from bus)	enabled	1 Bit	C-WTU	[1.18] DPT_Occupancy	258
Thermostat – Card Holder contact (from bus)	enabled	1 Bit	C-WTU	[1.18] DPT_Occupancy	259
Thermostat – Anticondensation (from bus)	enabled	1 Bit	C-WTU	[1.1] DPT_Switch	260

7.7.6 Weighted temperature value

The *Weighted temperature value* folder allows calculating a weighted mean average between the integrated temperature sensor value and the temperature sensor from bus value. The calculated temperature is then used as main temperature value in the integrated room temperature controller's algorithm.

The folder is active if: *Sensors from bus* ⇒ Room temperature (from bus) = enabled.

Parameter name	*Conditions*	*Values*
Relative weight		100% main sensor 90% / 10% 80% / 20% 70% / 30% 60% / 40% **50% / 50%** 40% / 60% 30% / 70% 20% / 80% 10% / 90% 100% sensor from bus
	The main sensor is always the integrated sensor; the sensor from bus needs to be activated in Sensors from bus folder.	
Minimum change of value to send [K]		**0,5** [other values in range 0 … 5 K]
	If the parameter is set to 0, no value is sent for change	
Cyclic sending interval		**no sending** [other values in range 30 s … 120 min]

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Weighted temperature	Cyclic sending interval ≠ no sending	2 Byte	CR-T--	[9.001] temperature °C	261

7.7.7 Heating

The *Heating* folder allows setting:

- The default value for single and relative setpoints (comfort, standby and economy setpoints);
- Control algorithm type (2-point hysteresis, PWM or continuous) and internal parameters;
- Building protection mode activation based on up to 2 window contacts' status.

Conditions of activation: *Settings* ⇒ Thermostat function = heating or both heating and cooling.

Parameter name	*Conditions*	*Values*
Temperature setpoint [°C]	Setpoint management = Single setpoint	**21** [range 10 … 50]
Comfort temp. setpoint [°C]	Setpoint management = Relative setpoints	**21** [range 10 … 50]
Standby temp. offset [0,1 K]	Setpoint management = Relative setpoints	**- 30** [range -10 … -50]
Economy temp. offset [0,1 K]	Setpoint management = Relative setpoints	**-50** [range -10 … -50]
Building protection temp. setpoint [°C]		**7** [range 2 … 10]
Control type		**2 point hysteresis**, PWM (pulse width modulation) continuous
Hysteresis	Control type = 2 point hysteresis	**0,3 K** [other values in range 0,2 K … 3 K]
Hysteresis position	Control type = 2 point hysteresis	**below** above
	The above hysteresis is suitable in case of special applications requiring mixing group control.	
Cyclic sending interval	Control type = 2 point hysteresis, continuous	**no sending** [other values in range 30 s … 120 min]
Min. change of value to send [%]	Control type = continuous	**10** [range 0 … 100]
PWM cycle time	Control type = PWM (pulse width modulation)	**15 min** [range 5 … 240 min]

Parameter name	*Conditions*	*Values*
Min control value [%]	Control type = PWM and continuous	**15 %** [range 0 %…30 %]
Max control value [%]	Control type = PWM and continuous	**85 %** [range: 70 %…100 %]
Heating type	Control type = continuous or PWM	**radiators (5 K / 150 min)** electric (4 K / 100 min) floor radiant panels (5 K / 240 min) ceiling radiant panels (5 K / 180 min) other
Proportional band [0,1 K]	Control type = continuous or PWM, Heating type = other	**40** [range 5 … 100]
Integral time [min]	Control type = continuous or PWM, Heating type = other	**90** [range 0 … 255 min]
Use window contacts to activate building protection mode	Sensors from bus ⇒ Window contact 1 = enabled, or Sensors from bus ⇒ Window contact 2 = enabled	**disabled /** enabled
	Parameter enabling window contact function.	
Waiting time before activating building protection mode	Use window contacts to activate building protection mode = enabled	**00:01:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Time interval before the device automatically switches to building protection mode.	

All communication objects are active if *Settings* ⇒ Thermostat function = heating or both heating and cooling.

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Input setpoint	Setpoint management = Single setpoint	2 Byte	CRWTU-	[9.001] temperature (°C)	271
Thermostat – Comfort setpoint (heating)	Setpoint management = Relative setpoints	2 Byte	CRWTU-	[9.001] temperature (°C)	271
Thermostat – Standby offset (heating)	Setpoint management = Relative setpoints, Heating/cooling switch over = manual	2 Byte	CRWTU-	[9.002] temperature difference (K)	273
Thermostat – Economy offset (heating)	Setpoint management = Relative setpoints, Heating/cooling switch over = manual	2 Byte	CRWTU-	[9.002] temperature difference (K)	275
Thermostat – Building protection setpoint (heating)		2 Byte	CRWTU-	[9.001] temperature (°C)	277
Thermostat – Standby setpoint (heating)	Setpoint management = Absolute setpoints	2 Byte	CRWTU-	[9.001] temperature (°C)	273
Thermostat – Economy setpoint (heating)	Setpoint management = Absolute setpoints	2 Byte	CRWTU-	[9.001] temperature (°C)	275

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Heating out command	Control type = 2 point hysteresis or PWM	1 Bit	CR-T--	[1.001] switch	280
Thermostat – Heating out command	Control type = continuous	1 Byte	CR-T--	[5.001] percentage (0..100%)	280

7.7.8 Cooling

The *Cooling* folder allows setting:

- The default value for single and relative setpoints (comfort, standby and economy setpoints) in case of manual heating/cooling switch over;
- The default value for the switch over dead band and for standby and economy attenuation in case of automatic heating/cooling switch over;
- Control algorithm type (2-point hysteresis, PWM or continuous) and internal parameters;
- Building protection mode activation based on up to 2 window contacts' status.

Conditions of activation: *Settings* ⇒ Thermostat function = heating or both heating and cooling.

Parameter name	*Conditions*	*Valori*
Temperature setpoint [°C]	Setpoint management = Single setpoint	**23** [campo 10 … 50]
Switch over dead band	Setpoint management = Relative setpoints, Heating/cooling switch over = automatic	**20** [campo 10 … 40]
Comfort temp. setpoint [°C]	Setpoint management = Relative setpoints, Heating/cooling switch over = from bus	**23** [campo 10 … 50]
Standby temp. offset [0,1 K]	Setpoint management = Relative setpoints	**30** [campo 10 … 50]
Economy temp. offset [0,1 K]	Setpoint management = Relative setpoints	**50** [campo 10 … 80]
Building protection temp. setpoint [°C]		**36** [campo 30 … 50]
Control type	Command for both heating and cooling = separated	**isteresi a 2 punti**, PWM (modulazione ad ampiezza d'impulso), continuo
	If in Settings the parameter Thermostat function = both heating and cooling and Command communication object = unique, the parameter Cooling type is bound to the choice done for Heating.	

Parameter name	*Conditions*	*Valori*
Hysteresis	Control type = 2 point hysteresis	**0,3 K** [altri valori nel campo 0,2 K … 3 K]
Hysteresis position	Control type = 2 point hysteresis	**superiore** inferiore
	The below hysteresis is suitable in case of special applications requiring mixing group control.	
Cyclic sending interval	Control type = 2 point hysteresis, continuous	**nessun invio** [altri valori nel campo 30 s … 120 min]
Min. change of value to send [%]	Control type = continuous	**10** [campo 0 … 100]
PWM cycle time	Control type = PWM (pulse width modulation)	**15 min** [campo 5 … 240 min]
Min control value [%]	Control type = PWM	**15** [campo 0…30]
Max control value [%]	Control type = PWM	**85** [campo 70…100]
Cooling type	Control type = continuous or PWM	pavimento radiante (5 K / 240 min) soffitto radiante (5 K / 180 min) altro
Proportional band [0,1 K]	Control type = continuous or PWM, Cooling type = other	**40** [campo 5 … 100]
Integral time [min]	Control type = continuous or PWM, Cooling type = other	**90** [campo 0 … 255 min]
Use window contacts to activate building protection mode	Sensors from bus ⇒ Window contact 1 = enabled, or Sensors from bus ⇒ Window contact 2 = enabled	**disabilitato** / abilitato
	Parameter enabling window contact function.	
Waiting time before activating building protection mode	Use window contacts to activate building protection mode = enabled	**00:01:00 hh:mm:ss** [campo 00:00:00 … 18:12:15]
	Time interval before the device automatically switches to building protection mode.	

All communication objects are active if *Settings* ⇒ Thermostat function = heating or both heating and cooling.

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Input setpoint	Setpoint management = Single setpoint	2 Byte	CRWTU-	[9.001] temperature (°C)	271
Thermostat – Comfort setpoint (cooling)	Setpoint management = Relative setpoints, Heating/cooling switch over = from bus	2 Byte	CRWTU-	[9.001] temperature (°C)	272

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Standby offset (cooling)	Setpoint management = Relative setpoints, Heating/cooling switch over = from bus	2 Byte	CRWTU-	[9.002] temperature difference (K)	274
Thermostat – Economy offset (heating)	Setpoint management = Relative setpoints, Heating/cooling switch over = from bus	2 Byte	CRWTU-	[9.002] temperature difference (K)	276
Thermostat – Building protection setpoint (cooling)		2 Byte	CRWTU-	[9.001] temperature (°C)	278
Thermostat – Standby setpoint (cooling)	Setpoint management = Absolute setpoints, Heating/cooling switch over = from bus	2 Byte	CRWTU-	[9.001] temperature (°C)	274
Thermostat – Economy setpoint (cooling)	Setpoint management = Absolute setpoints, Heating/cooling switch over = from bus	2 Byte	CRWTU-	[9.001] temperature (°C)	276
Thermostat – Cooling out command	Control type = 2 point hysteresis or PWM	1 Bit	CR-T--	[1.001] switch	281
Thermostat – Cooling out command	Control type = continuous	1 Byte	CR-T--	[5.001] percentage (0..100%)	281

7.7.9 Main and auxiliary ventilation

The ***Ventilation*** folder includes the following parameters:

- Ventilation function
- Control type
- Threshold first speed [0,1 K]
- Threshold second speed [0,1 K]
- Threshold third speed [0,1 K]
- Speed control hysteresis [K]
- Speed proportional band [0,1 K]
- Min. change of value to send [%]
- Manual operation
- Coil temperature usage for fan activation (Hot start)
- Antistratification function
- Disable ventilation from bus
- Signal from bus
- Fan start delay
- Fan stop delay

The conditions for the appearance of the ***Ventilation*** folder are:

Heating ⇒ Heating type = fan-coils or Cooling type = fan-coils

or a combination of the two conditions:

Heating ⇒ Heating type = floor radiant panels or ceiling radiant panels and *Heating* ⇒ Ventilation for auxiliary heating = enabled

Cooling ⇒ Cooling type = floor radiant panels or ceiling radiant panels and *Cooling* ⇒ Ventilation for auxiliary cooling = enabled

This way two types of installations can be controlled: i) fancoil terminals or ii) radiant panels as main stage and fancoil terminals as auxiliary stage.

7.7.9.1 Parameter and communication object tables

Parameter name	***Conditions***	***Values***
Control type		**1 speed** 2 speeds 3 speeds continuous regulation
Threshold first speed [0,1 K]	Control type ≥ 1 speed	**0** [range 0 … 255]
	The value is represented in tenths of Kelvin degrees. If the parameter Thermostat function = both heating and cooling, the threshold value is valid for both seasonal modes.	
Threshold second speed [0,1 K]	Control type ≥ 2 speeds	**10** [range: 0 … 255]
	The value is represented in tenths of Kelvin degrees. If the parameter Thermostat function = both heating and cooling, the threshold value is valid for both seasonal modes. For a correct operation of the ventilation, Threshold second speed > Threshold first speed.	

Parameter name	*Conditions*	*Values*
Threshold third speed [0,1 K]	Control type = 3 speeds	**20** [range 0 … 255]
	The value is represented in tenths of Kelvin degrees. If the parameter Thermostat function = both heating and cooling, the threshold value is valid for both seasonal modes. For a correct operation of the ventilation, Threshold third speed > Threshold second speed.	
Speed control hysteresis	Control type = 1, 2 or 3 speeds	**0,3 K** [other values in the range 0,2 K …3 K]
Speed proportional band [0,1 K]	Control type = continuous regulation	**30** [range 5 … 100]
	The value is represented in tenths of Kelvin degrees. If the parameter Thermostat function = both heating and cooling, the threshold value is valid for both seasonal modes.	
Min. change of value to send [%]	Control type = continuous regulation	**10** [range 2 … 40]
	Please refer to the Control Algorithms chapter for further information about the meaning of this parameter.	
Manual operation		**not depending on the temperature** depending on the temperature
	If the parameter = not depending on the temperature, the fan speed set by the user is not changed even when the temperature setpoint is reached; if the parameter = depending on the temperature, the fan stops when the temperature setpoint is reached.	
Hot start	Thermostat function = heating or both heating and cooling, Heating type = fancoils External sensors (from bus) ⇒ coil temperature = enabled	**no** / yes
	To carry out the function, a sensor for measuring the temperature of the heat exchanger of the fan coil has to be enabled. To this purpose, an external sensor (from bus) can be used.	
Min. temp.to start ventilation [°C]	Hot start = yes	**35** [range 28 …40]
	If enabled, the function is active only in heating mode.	
Antistratification function	External sensors (from bus) ⇒ Antistratification temperature = enabled	**disabled** / enabled
	To carry out the function, at least a sensor for measuring a second temperature value must be enabled at a different height than that of the room temperature controller. To this purpose, an external sensor (from bus) can be used.	
Antistratification temp. differential	Antistratification function = enabled	**2 [K/m]** [other values in the range 0,25 … 4,00 K/m]
	The DIN 1946 recommends a max temperature gradient of 2 K/m for rooms with standard height (between 2,70 and 3 m).	
Hysteresis	Antistratification function = enabled	**0,5 K** [other values in the range 0,2 … 3 K]
Disable ventilation from bus		**no** / yes
Signal from bus	Disable ventilation from bus = yes	**not inverted** inverted

Parameter name	*Conditions*	*Values*
Fan start delay		**0 s** [other values in the range 10 s … 12 min]
	This parameter is also available if the hot-start function is active (through measuring of the conveying fluid temperature at the battery for the thermal exchange). The function is active in both seasonal modes (heating and cooling).	
Fan stop delay		**0 s** [other values in the range 10 s … 12 min]
	The function allows prolonging the operation of the ventilator, dissipating in the room the residual heat or cool present in battery for the thermal exchange. The function is active in both seasonal modes (heating and cooling).	
Cyclic sending interval		**no sending** [other values in the range 30 s … 120 min]

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*Comm. Obj. No.*
Thermostat – Fan continuous speed	Control type = continuous regulation	1 byte	CR-T-	[5.001] percentage (0..100%)	286
Thermostat – Fan speed 1	Control type = 1, 2 or 3 speeds	1 bit	CR-T-	[1.001] switch	287
Thermostat – Fan speed 2	Control type = 2 or 3 speeds	1 bit	CR-T-	[1.001] switch	288
Thermostat – Fan speed 3	Control type = 3 speeds	1 bit	CR-T-	[1.001] switch	289
Thermostat – Fan control disable	Disable ventilation from bus = yes	1 bit	C-W--	[1.002] boolean	290
Thermostat – Fan manual speed		1 byte	CRW-U	[5.010] counter pulses (0…255)	293
Thermostat – Fan speed		1 byte	CR-T-	[5.010] counter pulses (0…255)	294
Thermostat – Fan manual active status		1 bit	CRWT-	[1.011] state	295
Thermostat – Fan manual speed percentage		1 byte	CR-T-	[5.001] percentage	310
Thermostat – Fan manual speed off status		1 bit	CR-T-	[1.011] state	311

7.7.9.2 Delayed fan start ("hot-start")

This function is used in case the fan forces in the room air passing through a heat exchange coil (as in the case of the terminals to the fan-coil). In the heating mode of operation, to avoid possible discomfort caused by the dispatch of cold air in the room, the room temperature controller does not start the fan until the fluid has not reached a sufficiently high temperature. This situation normally occurs at the first start or after long periods of inactivity. The function can be carried out by:

1. a temperature control (through a temperature sensor on the coil exchange battery);
2. a delayed start (function approximated);

In the first case the temperature of the heat conveying fluid is acquired at the exchange battery. The function then has an effective temperature control, but for the execution is necessary that the heat exchange coil is equipped with a sensor of minimum water temperature that acquires the temperature of the heat conveying fluid.

The effectiveness of the function depends on a field measurement of the time actually required to have sufficiently warm air from the terminal.

7.7.9.3 Antistratification function

This function is used in the case of heating systems with thermal exchange of convective type for rooms with height and volume much higher than usual (atriums, fitness facility, commercial buildings, etc.). Because of the natural convection - with warm air rising to the highest altitudes of the room - the phenomenon of air stratification occurs, with energy waste and discomfort for the occupants at the same time. The function opposes to the air stratification, forcing the warm air downwards.

The antistratification function requires:

- rooms of great height;
- availability of ventilation devices able to force the air movement downwards (opposed to the natural convective movement of warm air);
- measuring of the temperature at two heights through the installation of a second temperature sensor at an adequate height in order to measure the actual air stratification (the main room temperature controller is supposed to be installed at 1.5 m).

For rooms with ordinary height (2,70 ÷ 3,00 m) the DIN 1946 standard recommends not to exceed 2 K/m in order to have an adequate comfort; this gradient may be bigger in higher rooms.

7.7.9.4 2-stage configuration with fan-coils as auxiliary stage

The fan-coil units may be used both as a main stage and secondary stage. As main stage they can be combined only to radiators as auxiliary stage. If, however, the main stage is done with (floor or ceiling) radiant panels, the fan-coils can be used as auxiliary stage. In the latter case they work in automatic mode with a configurable offset with respect to the temperature setpoint for the main stage, and then carry out their compensation function while the main stage is brought in temperature with bigger inertia.

The ***Ventilation*** folder, that is unique, configures a main or a auxiliary stage depending on the settings choosed in the ***Heating*** and ***Cooling*** folders. Similarly, the display interface will act on manual / automatic and manual forcing of the only fan-coil.

A particular case occurs when a fan-coil unit works in a season as auxiliary stage and in the other one as main stage. It is for example the case of:

- a radiant panels system that works only for heating and has a fan-coil as auxiliary stage; the same fan-coil works as main stage for cooling;

- a radiator system that has a fan-coil as auxiliary stage for heating; the same fan coil unit functions as main stage for cooling.

In these cases with the configuration adopted, the following steps are necessary:

1. Settings ⇒ Thermostat function = both heating and cooling. This configuration enables both folders (heating and cooling)
2. Heating ⇒ Heating type = floor radiant panels or ceiling radiant panels
3. Heating ⇒ Command communication object = separated (if unique is choosen, the parameter Cooling ⇒ Cooling type does not appear)
4. Heating ⇒ Auxiliary heating = enabled
5. Auxiliary heating ⇒ Communication object = separated
6. Heating ⇒ Ventilation for auxiliary heating = enabled
7. Cooling ⇒ Cooling type = fancoils

Important! If the fan-coil system has a 2-pipe hydraulic configuration, the objects Auxiliary heating output command (1 bit) and Cooling out command (1 bit) have to be set in logical OR in the actuator for controlling the fan-coil which in this case is unique.

An alternative solution that avoids the setting of a logic OR can be realized by configuring a main stage for heating and cooling with radiant panels through separate valves and an auxiliary stage for heating and cooling fan coil through combined valves. The offset of the auxialiary stage for cooling is set to the value 0 (zero); this corresponds to a configuration for main stage. The object Cooling out command (1 byte) is not connected so that the radiant panel system works only for heating.

7.7.9.5 Remote fan speed modification

The communication objects shown in Figure 1 allow to monitor the actual fan speed forced automatically by the temperature controller or or by means of a supervisor system. The communication objects (from now on: C.O.) also allow to perform the same modifications remotely, for example from a supervisor software.

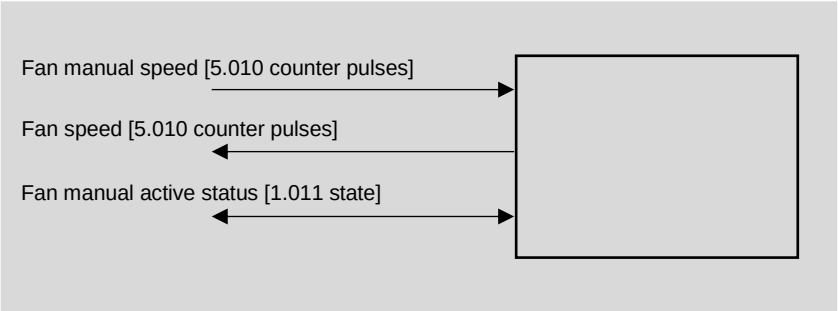


Figure 1 - Remote fan speed modification objects

The C.O. *Fan speed* allows to evaluate the actual fan speed; the C.O. *Fan manual active status* contains the information about automatic (=0, not active) or manual (=1, active) operating mode. By modifying the C.O. *Fan manual speed*, the fan automatically switches to the setpoint speed; to return to automatic mode (A), the supervisor must exit from manual mode by modifying the C.O. *Fan manual active status* (=0, not active).

Accepted values for C.O.s depend on the number of speeds set in ETS for the Ventilation Control type.

If *Control Type* parameter in Ventilation folder is = 1, 2 or 3 speeds, C.O.s with DPT [5.010 counter pulses] accept the following values:

- = 0: OFF
- = 1: speed 1
- = 2: speed 2 (if *Control Type* > 1 speed)
- = 3: speed 3 (if *Control Type* > 2 speed)

If *Control Type* parameter in Ventilation folder is = continuous regulation, the values of the C.O.s with DPT [5.010 counter pulses] match the following percentage of the maximum speed:

- = 0: OFF
- = 1: 20%
- = 2: 40%
- = 3: 60%
- = 4: 80%
- = 5: 100%

7.7.10 Relative humidity control

The ***Relative humidity control*** folder includes the following secondary folders:

- Dehumidification
- Humidification
- Calculated psychrometric values

The secondary folders ***Dehumidification***, ***Humidification*** and ***Calculated psychrometric values*** appear only if an external (from bus) humidity sensor is enabled.

The sensor acquires the air humidity value inside the room, which can be used for the following purposes:

- Sending over the bus (for information purpose) through DPT [9.007] percentage (%);
- Use of detected value for dew-point temperature calculations and sending on the bus through corresponding DPTs;
- Use for room ventilation through ventilation start, external intakes opening, window opening through motorized actuators. Control is performed upon thresholds;
- Use for control of thermoigrometric comfort conditions of radiant panel cooling systems equipped with integration of latent heat (starting of dedicated terminals without modification of cooling water flow temperature);
- Use for safety control in radiant panel cooling systems not equipped with integration of latent heat through calculation of critical thermoigrometric conditions (dew point) and corresponding modification of cooling water flow temperature.

Dehumidification

The secondary folder ***Dehumidification***, when the related function is enabled, includes the following parameters:

- Operating modes where dehumidification is active
- Relative humidity setpoint for dehumidification control [%]
- Dehumidification control hysteresis [%]
- Cyclic sending interval
- Disable dehumidification control from bus
- Signal from bus
- Dehumidification subordinated to temperature control
- Dehumidification start delay
- Sensible heat integration function
- Temperature difference for integration
- Hysteresis for integration

7.7.10.1 Parameter and communication object tables

Parameter name	*Conditions*	*Values*
Dehumidification function	Temperature control ⇒ Settings ⇒ Thermostat function = both heating and cooling	**disabled** cooling heating both cooling and heating
	Temperature control ⇒ Settings ⇒ Thermostat function = heating	**disabled** / heating only
	Temperature control ⇒ Settings ⇒ Thermostat function = cooling	**disabled** / cooling only
	Parameter that selects the dehumidification function.	
Humidity setpoint [%]	Dehumidification function ≠ disabled	**55** [range 20 … 80]
Humidity hysteresis	Dehumidification function ≠ disabled	**0,8 %** [other values in the range 0,5 … 4%]
Cyclic sending interval	Dehumidification function ≠ disabled	**no sending** [other values in the range 30 s … 120 min]
Disable dehumidification control from bus	Dehumidification function ≠ disabled	**no** / yes
Signal from bus	Disable dehumidification control from bus = yes	**not inverted** / inverted
Subordinated to temperature control	Temperature control ⇒ Settings ⇒ Thermostat function = cooling or both heating and cooling, Temperature control ⇒ cooling ⇒ cooling type = floor radiant panels or ceiling radiant panels, Relative humidity ⇒ dehumidification ⇒ dehumidification function = cooling only	no / **yes**
Dehumidification start delay	Subordinated to temperature control = no	**00:05:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Value 00:00:00 means that the start delay is disabled.	
Integration		**no** / yes
Temperature difference for integration	Integration = yes	**1,5°C** [other values in the range 0,5 °C … 3 °C]
Hysteresis for integration	Integration = yes	**0,5 K** [other values in the range 0,2 K … 3 K]

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*Comm. Obj. No.*
Thermostat – Relative humidity setpoint for dehumidification		2 bytes	CRWTU	[9.007] humidity (%)	299

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*Comm. Obj. No.*
Thermostat – Dehumidification command		1 bit	CR-T-	[1.001] switch	301
Thermostat – Dehumidification water battery command	Temperature control ⇒ Settings ⇒ Thermostat function = both heating and cooling, Temperature control ⇒ cooling ⇒ cooling type = floor radiant panels or ceiling radiant panels, Relative humidity control ⇒ dehumidification ⇒ dehumidification function = cooling only	1 bit	CR-T-	[1.001] switch	302
Thermostat – Dehumidification integration control	Temperature control ⇒ Settings ⇒ Thermostat function = both heating and cooling, Temperature control ⇒ cooling ⇒ cooling type = floor radiant panels or ceiling radiant panels, Relative humidity control ⇒ dehumidification ⇒ dehumidification function = cooling only Integration = yes	1 bit	CR-T-	[1.001] switch	303
	This object switches ON if (simultaneously) the relative humidity is greater than the relative humidity setpoint and the room temperature is greater than the setpoint of the parameter Temperature difference for integration.				
Thermostat – Dehumidification control disable	Disable dehumidification control from bus = yes	1 bit	C-W--	[1.002] boolean	304

Humidification

The secondary folder ***Humidification*** includes the following parameters:

- Operating modes where humidification is active
- Relative humidity setpoint for humidification control [%]
- Dehumidification control hysteresis [%]
- Cyclic sending interval
- Disable from bus

7.7.10.2 Parameter and communication object tables

Parameter name	***Conditions***	***Values***
Humidification function	Temperature control ⇒ Settings ⇒ Thermostat function = both heating and cooling	**disabled** cooling heating both cooling and heating
	Temperature control ⇒ Settings ⇒ Thermostat function = heating	**disabled** / heating only
	Temperature control ⇒ Settings ⇒ Thermostat function = cooling	**disabled** / cooling only
	Parameter that selects the humidification function.	
Humidity setpoint [%]	Humidification function ≠ disabled	**35** [range 20 … 80 %]
Humidity hysteresis [%]	Humidification function ≠ disabled	**0,8 %** [other values in the range 0,5 … 4%]
Cyclic sending interval	Humidification function ≠ disabled	**no sending** [other values in the range 30 s … 120 min]
Disable humidification control from bus	Humidification function ≠ disabled	**no** / yes
Signal from bus	Humidification function ≠ disabled Disable humidification control from bus = yes	**not inverted** / inverted

Object name	***Conditions***	***Dim.***	***Flags***	***DPT***	***Comm. Obj. No.***
Thermostat – Relative humidity setpoint for humidification	Humidification function ≠ disabled	2 bytes	CRWTU	[9.007] humidity (%)	300
Thermostat – Humidification command	Humidification function ≠ disabled	1 bit	CR-T-	[1.001] switch	305
Thermostat – Humidification control disable	Disable humidification control from bus = yes	1 bit	C-W--	[1.002] boolean	306

Calculated psychrometric values

The secondary folder ***Calculated psychrometric values*** includes the following parameters:

- Dew-point temperature
- Cyclic sending interval
- Min. change of value to send [K]

7.7.10.3 Parameter and communication object tables

Parameter name	***Conditions***	***Values***
Dew-point temperature		**disabled** / enabled
	The dew-point temperature, if sent on the bus, allows to implement an active anticondensation protection with recalibration of the flow conditions of the conveying fluid if each mixing group has its own control device. If the thermostat is installed in an environment where no air conditioning is foreseen (e.g. toilets), it is better to exclude that environment from the control by disabling the dew-point temperature parameter.	
Cyclic sending interval	Dew-point temperature = enabled	**no sending** [other values in the range 30 s … 120 min]
Min. change of value to send [K]	Dew-point temperature = enabled	**0,2 K** / no sending [other values in the range 0,2 … 3 K]

Object name	***Conditions***	***Dim.***	***Flags***	***DPT***	***Comm. Obj. No.***
Thermostat – Dew-point temperature	Dew-point temperature = enabled	2 bytes	CR-T-	[9.001] temperature °C	298

7.7.11 Energy saving

In order to implement energy-saving functions, window contacts (to detect the opening of windows or doors), presence and movement sensors and card holders can be used.

The ***Energy saving*** folder includes the following secondary folders:

- Window contacts
- Presence sensors
- Card holder

The folder is available if the following conditions are satisfied:

- *Internal sensors ⇒ Temperature sensor = enabled, or*
- *External sensors (from bus) ⇒ Room Temperature sensor = enabled.*

Window contacts

The ***Window contacts*** secondary folder appears if at least one sensor dedicated to this function is enabled i.e. if the following condition is verified:

- *External sensors (from bus) ⇒ Windows contact sensor 1 and/or 2 = enabled.*

The ***Window contacts*** folder includes the following parameters:

- Window contacts function
- Wait time to building protection mode

7.7.11.1 Parameter and communication object tables

Parameter name	*Conditions*	*Values*
Window contacts function		**disabled** / enabled
	This parameter enables the window contact function.	
Wait time to building protection mode	Window contacts function = enabled	**00:01:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Time interval before the automatic switching of the device to the Building protection operating mode	

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*Comm. Obj. No.*
Thermostat – Windows contact sensor 1 (from bus)	Window contacts function = enabled	1 bit	C-WTU	[1.019] window/door	255
Thermostat – Windows contact sensor 2 (from bus)	Window contacts function = enabled	1 bit	C-WTU	[1.019] window/door	256

Presence sensors

The ***Presence sensors*** folder includes the following parameters:

- Presence sensors function
- Presence sensors use
- Thermostat modes
- Absence time to switch HVAC mode

For this function external sensors (from bus) can be used, such as the ekinex EK-SM2-TP movement sensor or the ekinex EK-DX2-TP (X = B, C, D, E), or EK-DF2-TP, EK-DG2-TP, EK-DH4-TP presence sensor.

The following condition has to be true:

- *External sensors (from bus)* ⇒ *Presence sensor 1 and/or Presence sensor 2 = enabled,* and
- *Settings* ⇒ *Setpoint type = absolute* or *relative*

7.7.11.2 Parameter and communication object tables

Parameter name	***Conditions***	***Values***
Presence sensors function		**disabled** / enabled
	Parameter that enables the presence sensor function.	
Presence sensors use	Presence sensor function = enabled	**comfort extension** comfort limitation comfort extension and comfort limitation
Thermostat modes	Presence sensor function = enabled, Presence sensors use = comfort extension and comfort limitation, or comfort limitation	**comfort-standby** comfort-economy
Absence time to switch HVAC mode	Presence sensor function = enabled	**00:01:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Time interval before the automatic switching of the operating mode set in the Thermostat modes parameter.	

Object name	***Conditions***	***Dim.***	***Flags***	***DPT***	***Comm. Obj. No.***
Thermostat – Presence sensor 1 (from bus)	Presence sensor function= enabled	1 bit	C-WTU	[1.018] occupancy	257
Thermostat – Presence sensor 2 (from bus)	Presence sensor function = enabled	1 bit	C-WTU	[1.018] occupancy	258

Card holder

The ***Card holder*** secondary folder appears only if the corresponding sensor is enabled, i.e. if the following condition is true:

- *External sensors (from bus) ⇒ Card holder contact = enabled*

The ***Card holder*** folder includes the following parameters:

- Card holder function
- On card insertion switch HVAC mode to
- Activation delay on card insertion
- On card removal switch HVAC mode to
- Activation delay on card removal

7.7.11.3 Parameter and communication object tables

Parameter name	***Conditions***	***Values***
Card holder function		**disabled** / enabled
	Parameter that enables the card holder function.	
On card insertion switch HVAC mode to (*)	Card holder function = enabled	none **comfort** standby economy
	This parameter defines to which operating mode the device should automatically switch, after inserting the card into the holder. *(*) Note: if Setpoint type is set to **single** in the "Temperature control" tab, this parameter is set to "none", since the operating modes are not managed.*	
Activation delay on card insertion	Card holder function = enabled	**00:00:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Time interval before the automatic switching to the new operating mode, after inserting the card into the holder.	
On card removal switch HVAC mode to (*)	Card holder function = enabled	none **standby** economy building protection
	This parameter defines to which operating mode the device should automatically switch, after removing the card from the holder. **) Note: if Setpoint type is set to **single** in the "Temperature control" tab, this parameter is set to "**Building protection**", since the operating modes are not managed.*	
Activation delay on card removal	Card holder function = enabled	**00:00:00 hh:mm:ss** [range 00:00:00 … 18:12:15]
	Time interval before the automatic switching to the new operating mode, after removing the card from the holder.	

Object name	***Conditions***	***Dim.***	***Flags***	***DPT***	***Comm. Obj. No.***
Thermostat – Card Holder contact (from bus)	Card holder function = enabled	1 bit	C-WTU	[1.018] occupancy	259

Note on card holder function

The information of card insertion/removal in/from a card holder allows you to directly control the temperature by means of the room thermostat, while sending the object value on the bus allows you to control other room functions with KNX (lighting, electrical loads, feedback status for the hotel reception, etc.) depending on the configuration done with ETS. The value of the setpoint temperature and the switching have to be defined with the hotel responsible in accordance with the target of energy saving and level of service to be offered to the guests.

Conventional (not KNX) card holder

With a conventional card holder the status (card present or absent) of a signal contact is detected through an input of the device configured as *[DI] card holder contact sensor*. This way you can detect only the insertion and extraction of the card, but it cannot be detected e.g. the access of users with different profiles (guests, service staff, maintenance workforce).

KNX card holder

With a KNX card holder you can differentiate the switching to be carried out; this is not resolved by the parameters of the room temperature controller, but through the definition of scenes that are received by the device. Depending on the available device, advanced functions are possible (e.g. different user profiles).

7.8 Logic functions

The pushbutton interfaces EK-ED2-TP and EK-E13-TP allow to use some useful logic functions (AND, OR, NOT and exclusive OR) in order to implement complex functions in the building automation system.

You can configure:

- 8 channels of logical functions
- 4 inputs for each channel

Each object value, if desired, can be individually inverted by inserting a NOT logic operator.

For each channel, a parameter *Delay after bus voltage recovery* is available: this parameter represents the time interval between the bus voltage recovery and the first reading of the input communication objects for evaluating the logic functions.

In case of uncorrect connection of the input communication object or electrical trouble on bus resulting in a failed input reading request, the logic output of the corresponding channel can be calculated by setting the input values to default.

The communicaton function representing the logic function output is sent on the bus on event of change; alternatively, a cyclic sending can be set.

7.8.1 Parameter and communication object tables

The following condition has to be true: *General* ⇒ *Logic functions* = enabled.

Parameter name	***Conditions***	***Values***
Logic function		**disabled** / enabled
Logic operation	Logic function = enabled	**OR** / AND / XOR
	XOR (eXclusive OR)	
Delay after bus voltage recovery		**00:00:04.000 hh:mm:ss.fff** [range 00:00:00.000 … 00:10:55.350]
	Time interval between the bus voltage recovery and the first reading of the input communication objects for evaluating the logic functions	
Output cyclic transmission delay		**no sending** [other value in range 30 s … 120 min]
	No sending means that the output state of the logic function is updated on the bus only on change. Different values imply cyclic sending on the bus of the output state.	
Logic object x		**disabled** / enabled
Negated	Logic object x = enabled	**no** / yes
	Negando lo stato logico dell'ingresso corrispondente, è possibile realizzare logiche combinatorie articolate. Esempio: Output=(NOT(Oggetto logico 1) OR Oggetto logico 2).	
Read at startup	Logic object x = enabled	**no** / yes
Default value	Logic object x = enabled	**none** / off / on

Object name	*Conditions*	*Dim.*	*Flags*	*DPT*	*Comm. Obj. No.*
Logic function X – Input 1	Logic function X = enabled Logic object 1 = enabled	1 Bit	C-W--	[1.001] switch	190, 195, 200, 205, 210, 215, 220, 225
Logic function X – Input 2	Logic function X = enabled Logic object 2 = enabled	1 Bit	C-W--	[1.001] switch	191, 196, 201, 206, 211, 216, 221, 226
Logic function X – Input 3	Logic function X = enabled Logic object 3 = enabled	1 Bit	C-W--	[1.001] switch	192, 197, 202, 207, 212, 217, 222, 227
Logic function X – Input 4	Logic function X = enabled Logic object 4 = enabled	1 Bit	C-W--	[1.001] switch	193, 198, 203, 208, 213, 218, 223, 228
Logic function X – Output	Logic function X = enabled	1 Bit	C-W--	[1.001] switch	194, 199, 204, 209, 214, 219, 224, 229

8 Appendix

8.1 Summary of KNX communication objects

The following list contains the KNX communication objects for all corresponding *Data Point Types* (DPT) defined by the application program according to the performed configurations.

The list is ordered by object number; if the same object is linked to different inputs, the first input or rocker is referenced.

Object name	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Technical alarm	1 bit	C-W--	[1.5] DPT_Alarm	1
Leds intensity percentage	1 byte	C-W--	[5.1] DPT_Scaling	3
Rocker xx – Lock function	1 Bit	C-W--	[1.3] DPT_Enable	5, 22, 39, 56, 73, 90, 107, 124
Rocker x – Switching status [type], object n*	*Vedi tabella A1*	CR-TU	*Vedi tabella A1*	6..13, 23.. 30, 40.. 47, 57.. 64, 74.. 81, 91.. 98, 108.. 115, 125.. 132
	** I numeri degli O.C. elencati sono riferiti al primo di questi 8 oggetti (per ciascuno degli ingressi); gli O.C. degli oggetti successivi sono sequenziali. Per ottenere il numero dell'O.C. per l'n-esimo oggetto, aggiungere semplicemente (n-1) ai numeri riportati.* *Es.: gli O.C. associati all'ingresso 3A hanno numeri a partire da 81. Il numero del 5°O.C. associato a tale ingresso sarà quindi 81+ (5-1) = 85.*			
Rocker xx – Switching command	1 Bit	CRWTU	[1.1] DPT_Switch	14, 31, 48, 65, 82, 99, 116, 133
Rocker xx – Dedicated stop command	1 Bit	CRWTU	[1.17] DPT_Trigger	14, 31, 48, 65, 82, 99, 116, 133
Rocker xx – Dimming up / down / stop command	4 Bit	CR-T-	[3.*] DPT_Control_Dimming, DPT_Control_Blinds	15, 32, 49, 66, 83, 100, 117, 134
Rocker xx – Stop – step up/down command	1 Bit	CR-T-	[1.7] DPT_Step	17, 34, 51, 68, 85, 102, 119, 136
Rocker xx – Move up / down command	1 Bit	CRWTU	[1.8] DPT_UpDown	18, 35, 52, 69, 86, 103, 120, 137
Rocker xx – Scene number	1 Byte	CR-T-	[17.1] DPT_SceneNumber [18.1] DPT_SceneControl	19, 36, 53, 70, 87, 104, 121, 138
LED X – First colour	1 Bit	CRWTU	[1.1] DPT_Switch	141, 143, 245, 147, 149, 151, 153, 155
LED X – Second colour	1 Bit	CRWTU	[1.1] DPT_Switch	142, 144, 146, 148, 150, 152, 154, 156
Logic function X – Input 1	1 bit	C-WTU	[1.1] DPT_Switch	190, 195, 200, 205, 210, 215, 220, 225
Logic function X – Input 2	1 bit	C-WTU	[1.1] DPT_Switch	191, 196, 201, 206, 211, 216, 221, 226

Object name	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Logic function X – Input 3	1 bit	C-WTU	[1.1] DPT_Switch	192, 197, 202, 207, 212, 217, 222, 227
Logic function X – Input 4	1 bit	C-WTU	[1.1] DPT_Switch	193, 198, 203, 208, 213, 218, 223, 228
Logic function X – Output	1 bit	CR-T-	[1.1] DPT_Switch	194, 199, 204, 209, 214, 219, 224, 229
Temperature value	2 bytes	CR-T-	[9.1] DPT_Value_Temp	240
Temperature threshold 1 – Switch	1 bit	CR-T-	[1.1] DPT_Switch	241
Temperature threshold 1 – Lock	1 bit	C-W--	[1.1] DPT_Switch	242
Temperature threshold 1 – Value (from bus)	2 bytes	C-W--	[9.1] DPT_Value_Temp	243
Temperature threshold 2 – Switch	1 bit	CR-T-	[1.1] DPT_Switch	244
Temperature threshold 2 – Lock	1 bit	C-W--	[1.1] DPT_Switch	245
Temperature threshold 2 – Value (from bus)	2 bytes	C-W--	[9.1] DPT_Value_Temp	246
Thermostat – Room temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	247
Thermostat – Humidity (2 bytes, from bus)	2 bytes	C-W--	[9.7] DPT_Value_Humidity	248
Thermostat – Humidity (1 byte, from bus)	1 byte	C-W--	[5.1] DPT_Scaling	249
Thermostat – Antistratification temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	250
Thermostat – Outdoor temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	251
Thermostat – Coil temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	252
Thermostat – Floor temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	253
Thermostat – Flow temperature (from bus)	2 bytes	C-WTU	[9.1] DPT_Value_Temp	254
Thermostat – Windows contact sensor 1 (from bus)	1 bit	C-WTU	[1.19] DPT_Window_Door	255
Thermostat – Windows contact sensor 2 (from bus)	1 bit	C-WTU	[1.19] DPT_Window_Door	256
Thermostat – Presence sensor 1 (from bus)	1 bit	C-WTU	[1.18] DPT_Occupancy	257
Thermostat – Presence sensor 2 (from bus)	1 bit	C-WTU	[1.18] DPT_Occupancy	258
Thermostat – Card Holder contact (from bus)	1 bit	C-WTU	[1.18] DPT_Occupancy	259
Thermostat – Anticondensation (from bus)	1 bit	C-WTU	[1.1] DPT_Switch	260
Thermostat – Weighted temperature	2 bytes	CR-T-	[9.1] DPT_Value_Temp	261
Thermostat – Heating/cooling status out	1 bit	CR-T-	[1.100] DPT_Heat_Cool	262
Thermostat – Heating/cooling status in	1 bit	C-W--	[1.100] DPT_Heat_Cool	263
Thermostat - HVAC mode in	1 byte	C-W--	[20.102] DPT_HVACMode	264
Thermostat - HVAC forced mode in	1 byte	C-W--	[20.102] DPT_HVACMode	265
Thermostat - HVAC mode out	1 byte	CR-T-	[20.102] DPT_HVACMode	266
Thermostat - HVAC manual mode	1 byte	C-WTU	[20.102] DPT_HVACMode	267
Thermostat – Chrono active status	1 bit	CR-T-	[1.11] DPT_State	268
Thermostat – Actual setpoint	2 bytes	CR-T-	[9.1] DPT_Value_Temp	269
Thermostat – Manual setpoint	2 bytes	C-W--	[9.1] DPT_Value_Temp	270
Thermostat – Input setpoint	2 bytes	CRWTU	[9.1] DPT_Value_Temp	271
Thermostat – Comfort setpoint (heating)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	271
Thermostat – Comfort setpoint (cooling)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	272
Thermostat – Standby setpoint (heating)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	273
Thermostat – Standby offset (heating)	2 bytes	CRWTU	[9.2] DPT_Value_Tempd	273

Object name	Size	Flags	DPT	No. Comm. Obj.
Thermostat – Standby setpoint (cooling)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	274
Thermostat – Standby offset (cooling)	2 bytes	CRWTU	[9.2] DPT_Value_Tempd	274
Thermostat – Economy setpoint (heating)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	275
Thermostat – Economy offset (heating)	2 bytes	CRWTU	[9.2] DPT_Value_Tempd	275
Thermostat – Economy setpoint (cooling)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	276
Thermostat – Economy offset (cooling)	2 bytes	CRWTU	[9.2] DPT_Value_Tempd	276
Thermostat – Building protection setpoint (heating)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	277
Thermostat – Building protection setpoint (cooling)	2 bytes	CRWTU	[9.1] DPT_Value_Temp	278
Thermostat – Room temperature controller status	1 bit	CR-T-	[1.3] DPT_Enable	279
Thermostat – Heating out command	1 bit	CR-T-	[1.1] DPT_Switch	280
Thermostat - Heating out command	1 byte	CR-T-	[5.1] DPT_Scaling	280
Thermostat - Heating and cooling out command	1 bit	CR-T-	[1.1] DPT_Switch	280
Thermostat - Heating and cooling out command	1 byte	CR-T-	[5.1] DPT_Scaling	280
Thermostat – Cooling out command	1 bit	CR-T-	[1.1] DPT_Switch	281
Thermostat - Cooling out command	1 byte	CR-T-	[5.1] DPT_Scaling	281
Thermostat – Auxiliary heating output command	1 bit	CR-T-	[1.1] DPT_Switch	282
Thermostat - Auxiliary heating and cooling output command	1 bit	CR-T-	[1.1] DPT_Switch	282
Thermostat – Auxiliary cooling output command	1 bit	CR-T-	[1.1] DPT_Switch	283
Thermostat – Auxiliary heating disable	1 bit	C-W--	[1.3] DPT_Enable	284
Thermostat – Auxiliary cooling disable	1 bit	C-W--	[1.3] DPT_Enable	285
Thermostat – Fan continuous speed	1 byte	CR-T-	[5.1] DPT_Scaling	286
Thermostat – Fan speed 1	1 bit	CR-T-	[1.1] DPT_Switch	287
Thermostat - Fan speed 2	1 bit	CR-T-	[1.1] DPT_Switch	288
Thermostat - Fan speed 3	1 bit	CR-T-	[1.1] DPT_Switch	289
Thermostat – Fan control disable	1 bit	C-W--	[1.3] DPT_Enable	290
Thermostat – Alarm text	14 bytes	CR-T-	[16.0] DPT_String_ASCII	291
Thermostat – Manual setpoint active status	1 bit	CRWTU	[1.11] DPT_State	292
Thermostat – Fan manual speed	1 byte	CRWTU	[5.10] DPT_Value_1_Ucount	293
Thermostat – Fan speed	1 byte	CR-T-	[5.10] DPT_Value_1_Ucount	294
Thermostat – Fan manual active status	1 bit	CRWT-	[1.11] DPT_State	295
Thermostat – Room temperature control alarm	1 bit	CR-T-	[1.5] DPT_Alarm	296
Thermostat – Dew-point temperature	2 bytes	CR-T-	[9.1] DPT_Value_Temp	298
Thermostat – Relative humidity setpoint for dehumidification	2 bytes	CRWTU	[9.7] DPT_Value_Humidity	299
Thermostat – Relative humidity setpoint for humidification	2 bytes	CRWTU	[9.7] DPT_Value_Humidity	300
Thermostat – Dehumidification command	1 bit	CR-T-	[1.1] DPT_Switch	301
Thermostat – Dehumidification water battery command	1 bit	CR-T-	[1.1] DPT_Switch	302
Thermostat – Dehumidification integration control	1 bit	CR-T-	[1.1] DPT_Switch	303
Thermostat – Dehumidification control disable	1 bit	C-W--	[1.3] DPT_Enable	304
Thermostat – Humidification command	1 bit	CR-T-	[1.1] DPT_Switch	305
Thermostat – Humidification control disable	1 bit	C-W--	[1.3] DPT_Enable	306

Object name	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Thermostat – Thermal generator lock	1 bit	C-W--	[1.5] DPT_Alarm	308
Thermostat – Building protection HVAC mode active	1 bit	CR-T-	[1.11] DPT_State	309
Thermostat – Fan manual speed percentage	1 byte	CR-T-	[5.1] DPT_Scaling	310
Thermostat – Fan manual speed off status	1 bit	CR-T-	[1.11] DPT_State	311

Table A1. Dimensions e DPT for C.O.s with independent inputs:

Size	*DPT*
1 bit	[1.001] switch
2 bit	[2.*] 1-bit controlled
1 byte unsigned value	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte percentage	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte signed value	[6.*] 8-bit signed value
2 bytes unsigned value	[7.*] 2-byte unsigned value
2 bytes signed value	[8.*] 2-byte signed value
2 bytes floating value	[9.*] 2-byte float value

8.2 Warning

- Installation, electrical connection, configuration and commissioning of the device can only be carried out by qualified personnel.
- Opening the housing of the device causes the immediate end of the warranty period.
- ekinex® KNX defective devices must be returned to the manufacturer at the following address:

EKINEX S.p.A. Via Novara 37, I-28010 Vaprio d'Agogna (NO) Italy.

8.3 Other information

- This application manual is aimed at installers, system integrators and planners
- For further information on the product, please contact the ekinex® technical support at the e-mail address: support@ekinex.com or visit the website www.ekinex.com
- KNX® and ETS® are registered trademarks of KNX Association cvba, Brussels