CONTROL YOUR LIVING SPACE

APPLICATION MANUAL
EK-CC2-TP
EK-CD2-TP

UNIVERSAL INTERFACE
2 – 4 DIN
2 _4 Out Led

Summary

Revisione	Modifiche	Data
1.0.0	Emission	04/09/2017

1 Scope of the document

This application manual describes the application details for the ekinex® contact interface version:

- EK-CC2-TP 2 Digital Inputs and 2 Led pilot outputs
- EK-CD2-TP 4 Digital Inputs and 4 Led pilot outputs

Item	File name (## = relase)	Version	Device rel.	Update
Product dataschhet	STEKCDG2TP_EN.pdf	EK-CC2-TP EK-CD2-TP	A1.0	09/2017
Application manual	MAEKCD2TP_EN.pdf	EK-CC2-TP EK-CD2-TP		
Application program	APEKCC2TP##.knxprod	EK-CC2-TP		
Application program	APEKCD2TP##.knxprod	EK-CC2-TP		

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

EK-CC2-TP:……..

EK-CD2-TP

2 Product description

The EK-CC2-TP modules and EK-CC2-TP ekinex® include four separate digital inputs each configurable as:
• Binary input

This device is equipped with an integrated communication module for KNX bus and is intended for box mounting; DIN rail or coupled to the FF and 71 series

The device also comes with programmable LED outputs for each command, which can be used for signaling functions or as nighttime orientation.

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

2.1 Input functions

Each one of two active positions of the input, or physical pushbutton, of the device. Such actions, in relation to a single input, will be labelled with letters A and B..

When the imput 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 Input 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 Input 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 Input can also be programmed to perform exactly the same operation, thereby effectively causing one Input to act as a single pushbutton..

2.2 Led Output

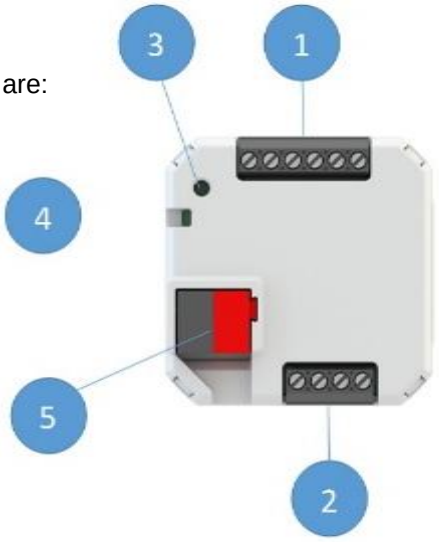
The interface has a number of outputs for the signal LED connection, the number of inputs, which can be freely programmed (also with functions independent of inputs), both as functional indications and for obtaining aesthetic effects or as night orientation lights.

For a more detailed description of the LED outputs and their configuration parameters refer to the application section of the manual.

2.3 Connection elements

The elements present and necessary for connecting the device are:

1. Digital inputs terminal block
2. Signal LED link terminal block
3. Aggregate button
4. LED programming mode indication
5. KNX bus line connection



3 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.
The application program allows you to access, within the ETS4 / 5 environment, the configuration of all the work parameters of the device.
The program must be loaded into ETS (alternatively you can only load the entire ecinex® database of products in one operation), and then all device specimens of the type considered can be added to the project being drafted.

Configurable parameters for the device will be described in detail in the following paragraphs.

The configuration can be, and will generally be, completely defined in off-line mode; the transfer to the configured device will then occur at the programming stage, described in next paragraph.

.

Device Code	Input n°	Out Led n°	Applicative Program ETS (## = version)	Cominication Objects (Nr. max)	Group Addresses (Nr. max)
EK-CC2-TP	2	2	APEKCC2TP##.knxprod	93	255
EK-CD2-TP	4	4	APEKCD2TP##.knxprod	93	255

4 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 operation to programming mode by pressing the programming pushbutton located on the rear side of the housing. In this mode of operation, the programming LED is turned on steady;
- upload the configuration (including the physical address) to the device with the ETS program.

At the end of the upload, the operation of the device automatically returns to normal mode; in this mode the programming LED is turned off. 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;
- reagire ai telegrammi sul bus di richiesta dello stato degli ingressi o delle variabili locali.
- 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) 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 Pushbotton 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 pairs

The 4 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 Input.

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 Input can be configured in two ways: single mode and coupled mode. This setting appears among Input-level settings rather than input-level settings, because only inputs belonging to the same Input 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 Input are effectively operated "in parallel", so as to operate the whole Input 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 Input, 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.

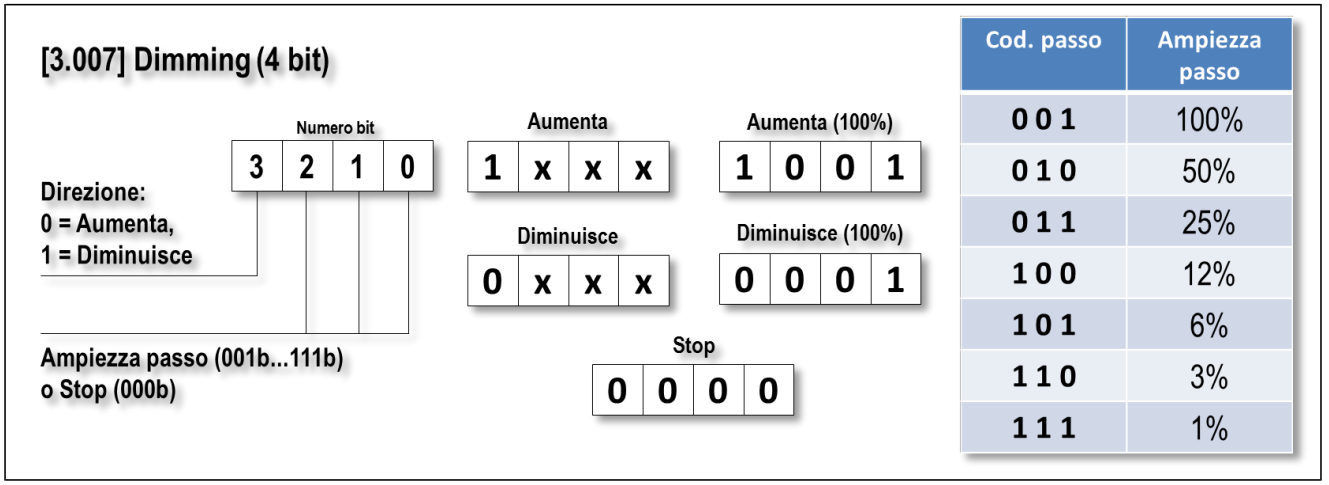
3. *Shutter or Venetian blind control*
   The two inputs of a pair are assigned to opposite movement directions; these can be assigned to inputs as desired, i.e. A up / B down or the other way around.
   The *blinds* mode can also be set, and it works exactly as in single mode.

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.

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 Input 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.

.

The defined operations and related commands are the following:

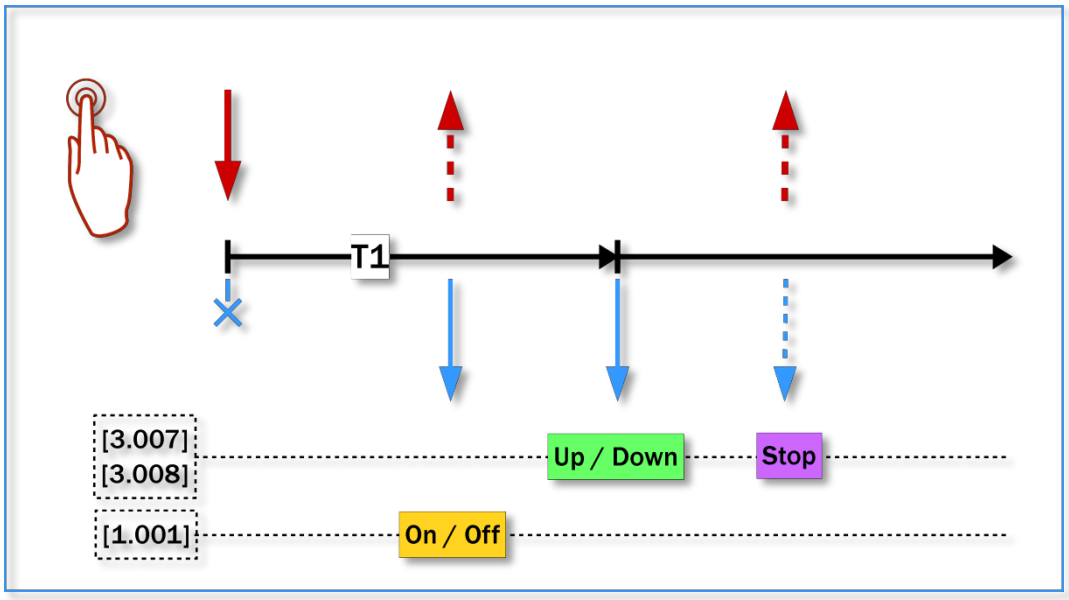


Figure 1 - Dimmer mode command sequence

- 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

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 Input 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.

.

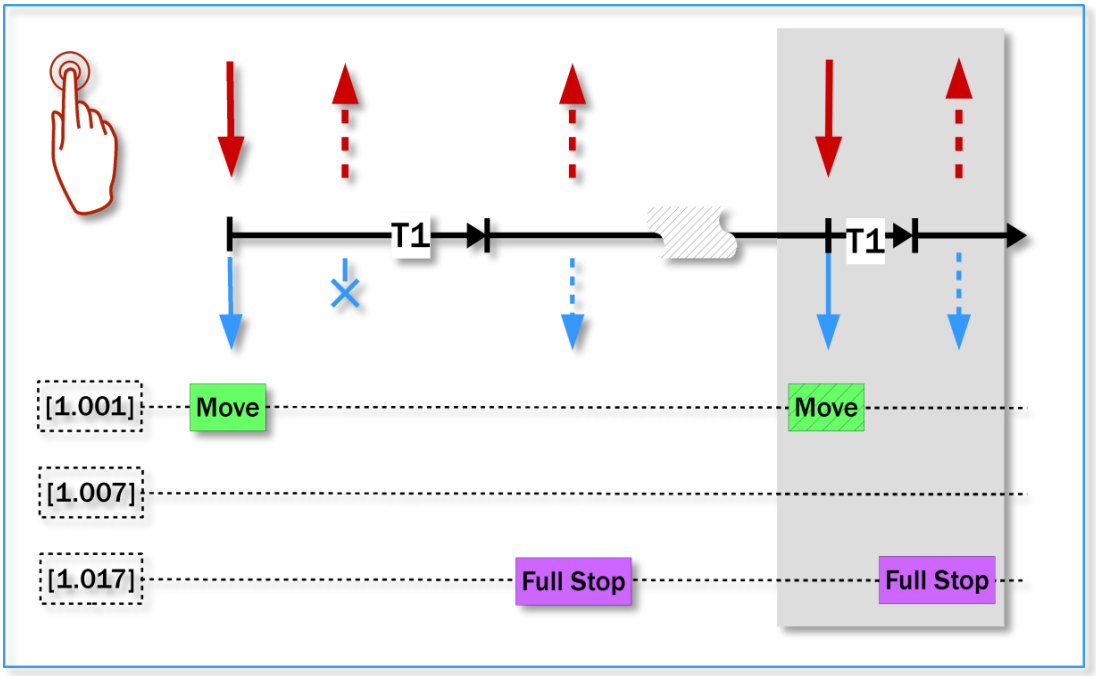


Figure 2 - “Shutter” mode command sequence

In “Shutter” mode, when a Input 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 Input is in independent mode and has been configured as *toggle*).
If the Input 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 Input with a long press.
If the Input is pressed with a long press, when it is released – which will be in correspondence with the desired position – the shutter will stop.

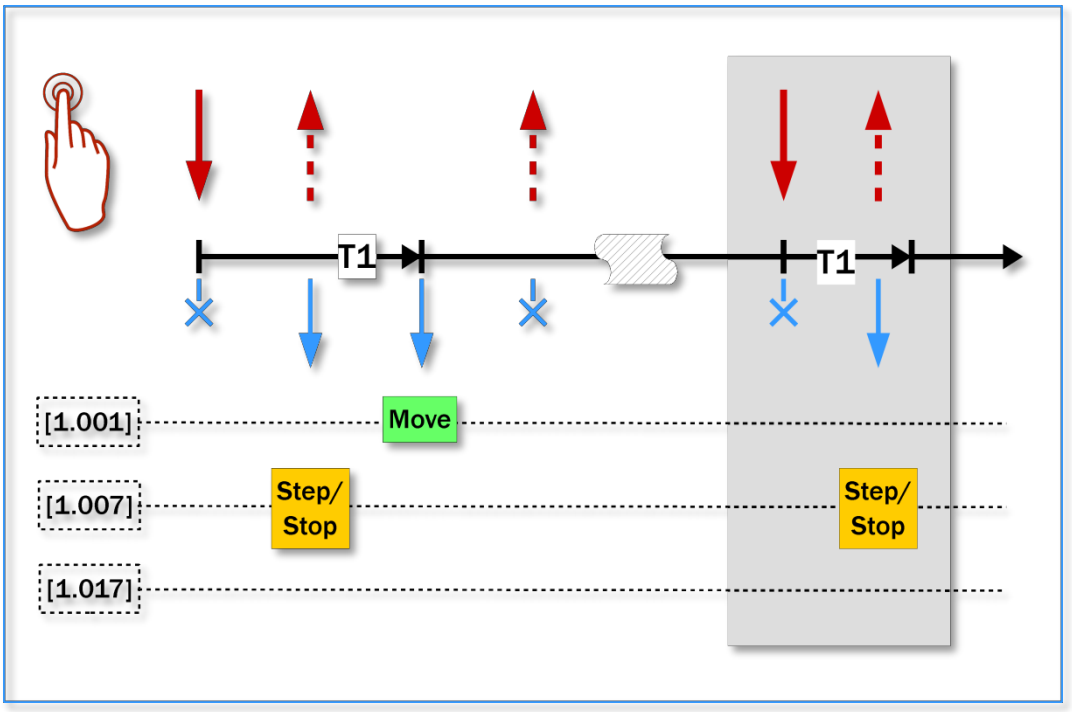


Figure 3 - “Venetian blind” mode command sequence

In “Venetian blind” mode, on release of a Input 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 Input 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 Input needs to be pressed again (short press).

6.5 Outputs for LED signaling

The LED indicators associated with each input can be individually addressed even if the corresponding inputs are paired.

6.5.1 Individual parameters.

The power of each LED can be set as follows:

- Fixed value (always on or off)
- Switches on when the corresponding input is activated. With this option, you can specify an additional delay after the button is released;
- Status determined by the bus through via communication object. In this case, you can specify that in the active state the LED is flashing (with different choices for on / off times); In addition, the on / off condition can be reversed with respect to the status of the reference communication object (LED lit when the value of the object is "off" and vice versa).

6.5.2 Funzioni logiche

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:

- 4 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:

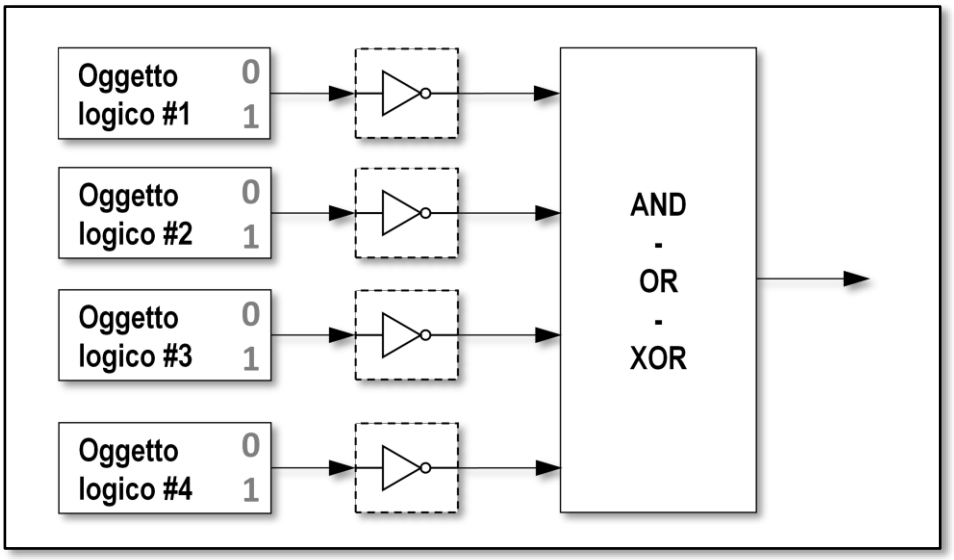


Figure 4 – Logic combination function

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:

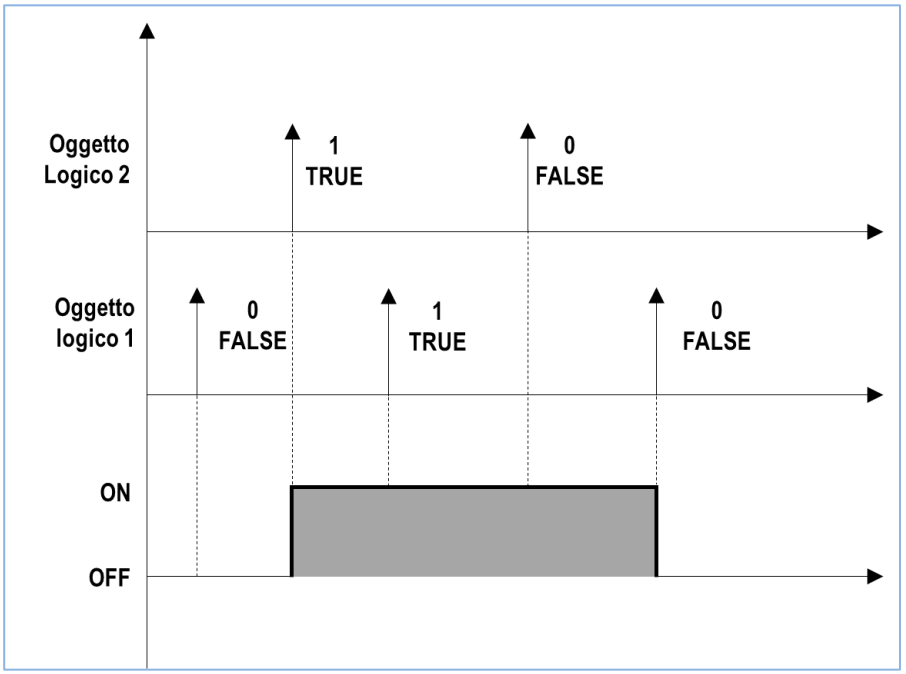


Figure 5 – Logic function OR

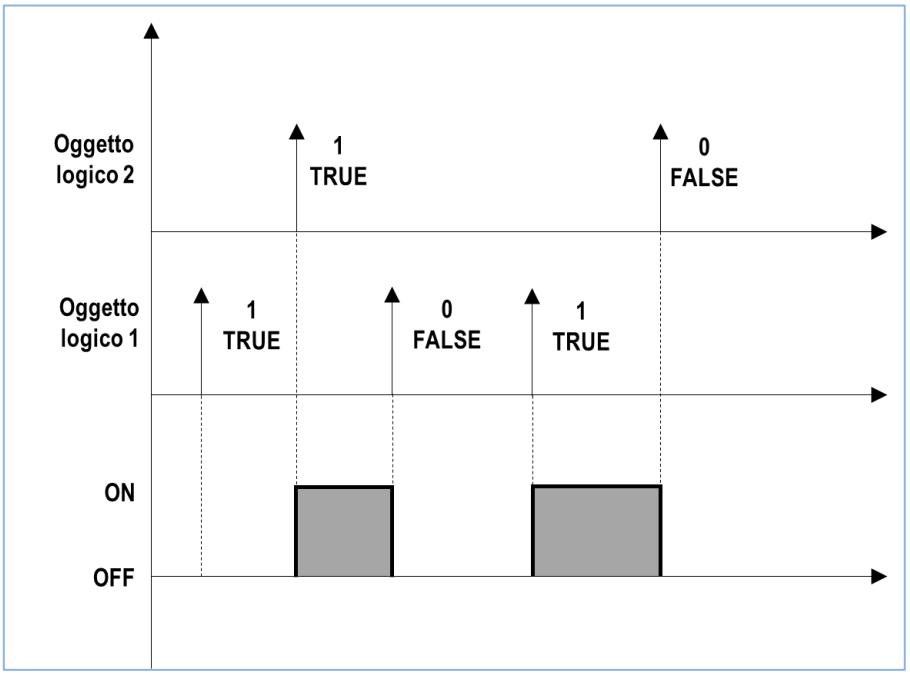


Figure 7 – Logic function AND

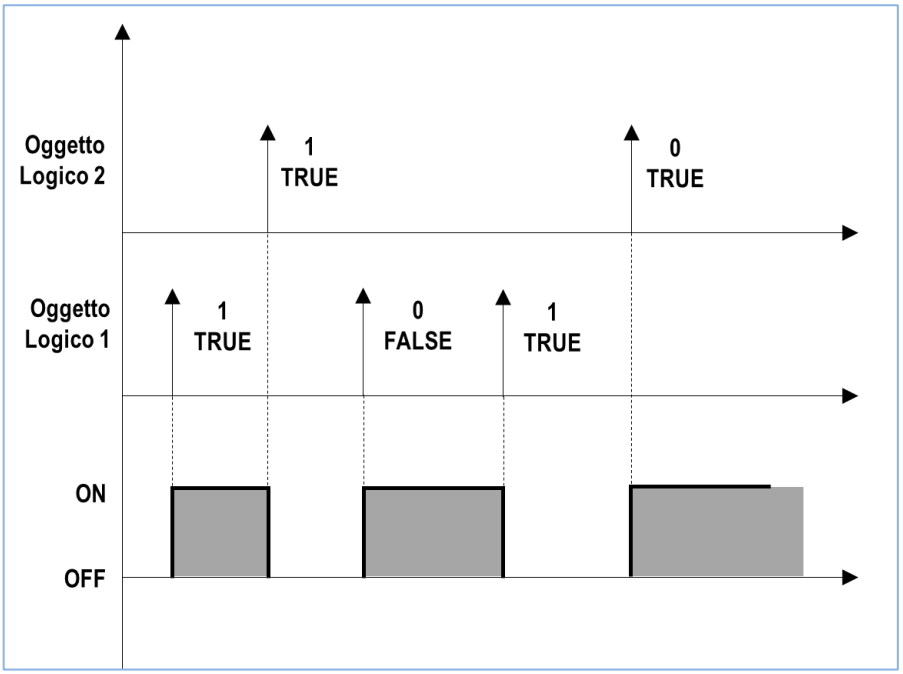


Figure 8 – Logic function XOR

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.

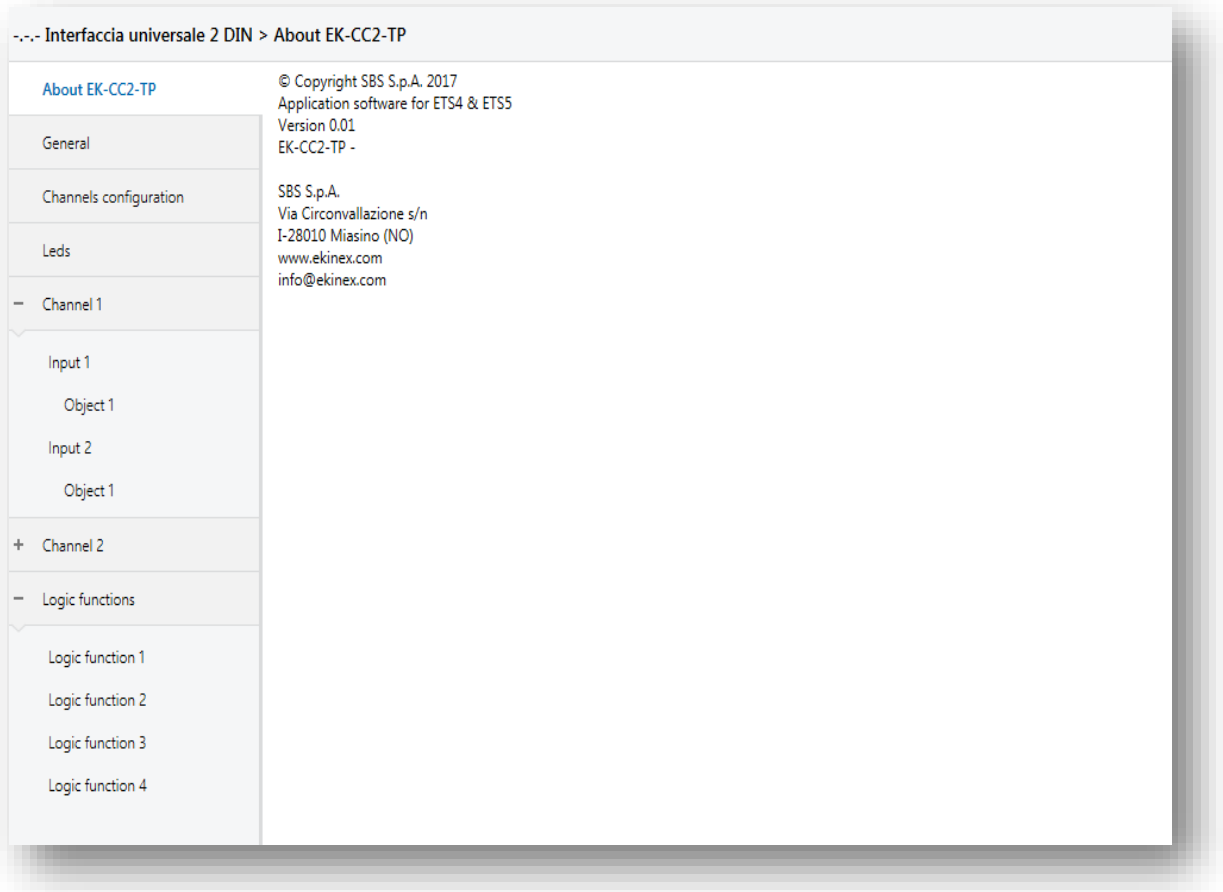
7 Application program for ETS

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…2), while a generic input is referenced as “xx” (xx = 1A, 1B, 2A, 2B).

I valori dei parametri evidenziati in neretto sono quelli di *default*.



7.1 Info su EK-EC2/ED2-TP

The folder ***About EK-CC2-TP*** and ***EK-CD2-TP*** is for information purposes only and does not contain parameters to be set. The information given is:

Application software per ETS4/5
Version 1.00
Universal Interface 2 o 4 DIN

SBS S.p.A.
Via Circonvallazione s/n
I-28010 Miasino (NO)
Italy www.ekinex.com
info@ekinex.com

7.2 General setting

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***
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.	
Logic functions		enabled / **disabled**
	Enables the folders to configure AND, OR e XOR logic functions and their relative input and output communication objects.	

7.3 Input configuration

Nome parametro	***Condizioni***	***Valori***
Input *x*	-	disabled **independent or single** coupled temperature probe
	Sets the operation mode for inputs associated with Input x. The identification of which inputs and physical inputs corresponding to a given number (eg 1A - 2B - 4A etc.), depending on the layout chosen for the plates, depends on the parameter "General / Input Configuration	
Function A	Input *x* = independent or single	**enabled /** disabled
	Enables or disables the capability to generate events for the first pushbutton of the input.	
Type	Input 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 Input input. *Further parameters for the selected function will appear in the individual Input configuration sections (see below).*	

Nome parametro	*Condizioni*	*Valori*
Temperature probe A	Input x = temperature probe Function A = enabled	enabled **/ disabled**
	Determines the kind of function performed by the FIRST Input input. *Further parameters for the selected function will appear in the individual Input configuration sections (see below).*	
Function B	Input x = independent or single	disabled enable in parallel with function A, as a function. single **copy parameters from function A**
	Enables or disables the ability to generate events for the second Input button. *If it is not disabled, the second button can be assigned its own independent function, it can be used as a "duplicate" of the first input (in parallel ...) or perform the same type of function as the first button but based on its own independent communication objects .*	
Type	Input 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 FIRST Input input. *Further parameters for the selected function will appear in the individual Input configuration sections (see below).*	
Type	Ingresso *x* = accoppiato	switch **dimming** blinds or Venetian blinds
	Determines the kind of function performed by the FIRST and SECOND Input input. *Further parameters for the selected function will appear in the individual Input configuration sections (see below).*	
Temperature probe B	Input x = temperature probe Function B = enabled	enabled **/ disabled**
	Determines the kind of function performed by the FIRST Input input. *Further parameters for the selected function will appear in the individual Input configuration sections (see below).*	

7.3.1 Independent or single: send values or sequences

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input x – Switching status [type], object n	Input 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**)	5, 22 (1A, 1B) 46,60 (2A, 2B)
	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 Input 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**.*				

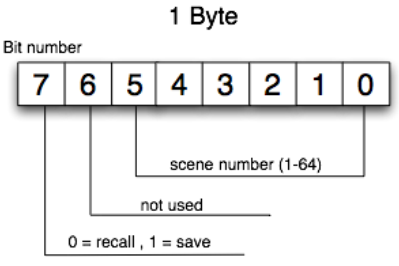
7.3.2 Independent or single: dimming

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input x – Switching command	Input x = independent or single Function x = enabled Type = dimming	1 bit	CRWTU	[1.001] switch	13,30 (1A, 1B) 51,68 (2A, 2B)
	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.*				
Input x – Dimming up / down / stop command	Input x = independent or single Function x = enabled Type = dimming	4 bit	CR-T-	[3.*] 3-bit control	14,31 (1A, 1B) 52,69 (2A, 2B)
	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: 1 0 0 0 — Decrease: 0 0 0 1 Stop dimming: 0 0 0 0 *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.*				

7.3.3 Independent or single: shutter or venetian blind

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input x – Dedicated stop command	Input x = independent or single Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.017] trigger	13,30 (1A, 1B) 51,68 (2A, 2B)
	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.				
Input x – Stop – step up/down command	Input x = independent or single Function x = enabled Type = shutter or venetian blind Blind mode = enabled	1 bit	CR-T-	[1.007] step	16,33 (1A, 1B) 54,71 (2A, 2B)
	Move the blind to fully open or fully closed position. The object is sent at the end of a long press.				
Input x – Move up / down command	Input x = independent or single Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.008] up/down	17,34 (1A, 1B) 55,72 (2A, 2B)
	Increase or decrease the opening of the blind stepwise. *The object is sent on a short press.*				

7.3.4 Independent or single: scene

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input x – Scene number	Input x = independent or single Function x = enabled Type = scene	1 Byte	CR-T-	[17.*] Scene number [18.*] Scene control	18,35 (1A, 1B) 61,73 (2A, 2B)
	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). 				

7.3.5 Coupled: switch

Object name	Conditions	Size	Flags	DPT	No. Comm. Obj.
Input x – Switching command	Input x = coupled Function x = enabled Type = switch	1-bit	CRWTU	[1.001] switch	13 51 89 127

7.3.6 Coupled: dimming

Object name	Conditions	Size	Flags	DPT	No. Comm. Obj.
Input x – Switching command	Input x = coupled Function x = enabled Type = dimming	1 bit	CRWTU	[1.001] switch	13 51 89 127
	See notes for independent input.				
Input x – Dimming up / down / stop command	Input x = independent or single Function x = enabled Type = dimming	4 bit	CR-T-	[3.*] 3-bit control	14 52 90 128
	See notes for independent input.				

7.3.7 Coupled: shutter or venetian blind

Object name	Conditions	Size	Flags	DPT	No. Comm. Obj.
Input x – Dedicated stop command	Input x = coupled Function x = enabled Type = shutter or venetian blind Blind mode = disabled	1 bit	CRWTU	[1.017] trigger	13 51 89 127
	See notes for independent input.				
Input x – Stop – step up/down command	Input x = coupled Function x = enabled Type = shutter or venetian blind Blind mode = enabled	1 bit	CR-T-	[1.007] step	16 54 92 130
	See notes for independent input.				
Input x – Move up / down command	Input x = coupled Function x = enabled Type = shutter or venetian blind	1 bit	CRWTU	[1.008] up/down	17 55 93 131
	See notes for independent input.				

7.4 *Input* x: Function A/B *configuration*

7.4.1 Indipendent 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	Input 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	Input 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.	

7.4.2 Independent or single: Lock function enabled

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input xx – Lock function	Input x = Independent or single Lock function = enabled	1 bit	C-W--	[1.003] enable	4,21 (1A, 1B) 42,59 (2A, 2B)

When the lock function is enabled, for each input or Input 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

<table>
<tr><th>Channel mode</th><th>Input type</th><th>Behaviour at locking</th><th>Behaviour at unlocking</th></tr>
<tr><td>independent
</td><td rowspan="2">send values or sequences</td><td colspan="2">none
as close or short press
as open or long press</td></tr>
<tr><td>
coupled</td><td rowspan="3">switching
dimming</td><td rowspan="3">none
off
on
toggle</td><td rowspan="3">none
off
on
as previous</td></tr>
<tr><td>independent</td></tr>
<tr><td>coupled</td></tr>
<tr><td>independent
</td><td rowspan="2">scene</td><td colspan="2">none
send first scene
send second scene</td></tr>
<tr><td>independent</td><td rowspan="2">shutter or venetian blind</td><td colspan="2" rowspan="2">none
up
down</td></tr>
<tr><td>coupled</td></tr>
</table>

\

7.4.3 Independent or single: send values or sequences

Parameter name	*Conditions*	*Values*
Number of communication objects	Input 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	Input 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	Input 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	Input 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	Input 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	Input 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	Input 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	Input 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	Input 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	Input x = independent or single Type = send values or sequences send dimension = 1 bit	none on off **toggle**
	Input 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

Parameter name	*Conditions*	*Values*
	Input 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	Input x = independent or single Type = send values or sequences send dimension = 1 bit	**none** on off toggle
	Input 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
	Input 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	Input 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) -671088.64...670760.96 (2 bytes floating value)
	First value available for association in send events	
Object n – Value 2	Input 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.*
Input xx – Switching status [type] Object n	Input x = Independent or single Type = send values or sequences	*see the table below*	CR-TU	*see the table below*	3..35 (1A, 1B) 36…69 (2A, 2B)

Le dimensioni dei dati e i *Data Point Types* sono i seguenti:

Dimens.	***DPT***
1 bit	[1.001] switch
2 bit	[2.*] 1-bit controlled
1 byte senza segno	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte percentuale	[4.*] character [5.*] 8-bit unsigned value [20.*] 1-byte
1 byte con segno	[6.*] 8-bit signed value
2 bytes senza segno	[7.*] 2-byte unsigned value
2 bytes con segno	[8.*] 2-byte signed value
2 bytes virgola mobile	[9.*] 2-byte float value

7.4.4 Independent or single: dimming

Parameter name	*Conditions*	*Values*
Long press time	Input 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	Input 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	Input 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	Input 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	Input 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	Input x = independent or single Type = dimming Send cyclically ≠ none	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Input 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	Input x = independent or single Type = dimming	**none** off on as previous
	Value to be assigned to the object when an unlocking command is received.	

7.4.6 Independent or single: scene

Parameter name	*Conditions*	*Values*
First scene number	Input 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	Input 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	Input 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	Input 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	Input 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	Input 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	Input x = independent or single Type = scene	**none** send first scene send second scene
	Operation to perform when an unlocking command is received.	

7.4.7 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	Input x = coupled	enabled / **disabled**
	Enables or disables the capability of locking the input through a remote command (telegram).	

7.4.8 Coupled: Lock function enabled

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*No. Comm. Obj.*
Input xx – Lock function	Input x = coupled Lock function = enabled	1 bit	C-W--	[1.003] enable	4 42

7.4.9 Coupled: switch

Parameter name	*Conditions*	*Values*
xA and xB use	Input x = coupled Type = switch	**A on, B off** A off, B on
	Allows to invert side A and side B functionalities	
Send cyclically	Input 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	Input x = coupled Type = switch Send cyclically ≠ none	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Input 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	Input x = coupled Type = switch	**none** on off as previous
	Value to be assigned to the object when an unlocking command is received.	

7.4.10 Coupled: dimming

Parameter name	*Conditions*	*Values*
Long press time	Input 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	Input x = coupled Type = dimming	**A increases, B decreases** A decreases, B increases
Send cyclically	Input 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	Input x = coupled Type = dimming Send cyclically ≠ no	hh:mm:ss (**00:02:00**)
	Interval between cyclical transmissions.	
Lock function – Behaviour at locking	Input 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	Input x = coupled Type = dimming	**none** on off as previous
	Value to be assigned to the object when an unlocking command is received.	

7.4.11 Coupled: shutter or venetian blind

Parameter name	*Conditions*	*Values*
Long press time	Input 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	Input x = coupled Type = shutter or venetian blind	**A up, B down** A down, B up
Blind mode	Input 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	Input x = coupled Type = shutter or venetian blind	**none** up down
	Operation to perform when a locking command is received.	
Lock function – Behaviour at unlocking	Input 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 *Inputs Configuration* section

8 Logic functions

The pushbutton interfaces EK-CC2-TP and EK-CD2-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:

- 4 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.

8.1Parameter 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	204,209,214,219
Logic function X – Input 2	Logic function X = enabled Logic object 2 = enabled	1 Bit	C-W--	[1.001] switch	205,210,215,220
Logic function X – Input 3	Logic function X = enabled Logic object 3 = enabled	1 Bit	C-W--	[1.001] switch	206,211,216,221
Logic function X – Input 4	Logic function X = enabled Logic object 4 = enabled	1 Bit	C-W--	[1.001] switch	207,212,217,222
Logic function X – Output	Logic function X = enabled	1 Bit	C-W--	[1.001] switch	208,213,218,223

9 Configurazione uscite LED di segnalazione

I seguenti parametri di configurazione sono da intendersi ripetuti per ciascuno dei LED disponibili.

Le impostazioni per i LED sono elencate sempre raggruppate per il corrispondente Ingresso, indipendentemente dal fatto che gli ingressi siano utilizzati in modalita accoppiata o meno.

Nome parametro	*Condizioni*	*Valori*
LED X	-	fixed **closed contact** was from the
Always	LED X = fixed	on / **off**
Off Delay	LED X = closed contact	hh:mm:ss.ff (**00:02:00.00**)
	LED Power Off Delay After Power On Term	
Blinking	LED X = from bus	**No** /Yes
Signal from Bus	LED X = from bus	**Not inverted** / inverted
	Specifies whether the state of the LED received from the bus should be interpreted inverted, ie es. LED lit when receiving an "off" command via a communication object. This feature is useful because the LED on can be linked to a communication about the status of other entities, which have an opposite logic.	

Nome oggetto	*Condizioni*	*Dimens.*	*Flags*	*DPT*	*Nr. Ogg. Com.*
LED X –	LED = from bus	1 bit	CRWTU-	[1.001] switch	70,71,72.73

10 Appendice

10.1 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 Input is referenced.

Num	*Object Name*	*Size*	*Flags*	*DataPoint Type*
1	TechnicalAlarm	1 Bit	-WC---	[1.1] DPT_Switch
2	Input 1A - LockCommand	1 Bit	-WC---	[1.1] DPT_Switch

3	Input 1A - SwitchingStatus1	Variabile(*)	RWCTU-	Variabile(**)
4	Input 1A - SwitchingStatus2	Variabile(*)	RWCTU-	Variabile(**)
5	Input 1A - SwitchingStatus3	Variabile(*)	RWCTU-	Variabile(**)
6	Input 1A - SwitchingStatus4	Variabile(*)	RWCTU-	Variabile(**)
7	Input 1A - SwitchingStatus5	Variabile(*)	RWCTU-	Variabile(**)
8	Input 1A - SwitchingStatus6	Variabile(*)	RWCTU-	Variabile(**)
9	Input 1A - SwitchingStatus7	Variabile(*)	RWCTU-	Variabile(**)
10	Input 1A - SwitchingStatus8	Variabile(*)	RWCTU-	Variabile(**)
11	Input 1A - SwitchingCommand-DedicatedStop	1 Bit	-WCTU-	[1.1] DPT_Switch
12	Input 1A - DimmingUpDownStopCommand	4 Bit	R-CT--	[3.7] [3.8] DPT_Dimming
13	Dummy - Input 1A - InfoStatus	1 Bit	-WC---	[1.1] DPT_Switch
14	Input 1A - StopStepUpDownCommand	1 Bit	R-CT--	[1.17] DPT_Trigger
15	Input 1A - MoveUpDownCommand	1 Bit	RWCTU-	[1.8] DPT_Up/Down
16	Input 1A - SceneNumberControl	1 Bytes	--CT--	[17.1] [18.1] DPT_ Scene
16	Input 1A – CounterValue 1 Byte	1 Bytes	R-CT--	[5.1] DPT_ Pulses
16	Input 1A – CounterValue 2 Byte	2 Bytes	R-CT--	[7.1] DPT _ Pulses
16	Input 1A – CounterValue 4 Byte	4 Bytes	R-CT--	[12.1] DPT_Counter pulses
17	Input 1A - Counter reset command	1 Bit	-WC-U-	[1.15] DPT_Reset
18	Input 1A - Counter runout	1 Bit	RWCTU-	[1.5] DPT_Alarm
19	Input 1B - LockCommand	1 Bit	-WC---	[1.3] DPT_Enable
20	Input 1B - SwitchingStatus1	Variabile(*)	RWCTU-	Variabile(**)
21	Input 1B - SwitchingStatus2	Variabile(*)	RWCTU-	Variabile(**)
22	Input 1B - SwitchingStatus3	Variabile(*)	RWCTU-	Variabile(**)
23	Input 1B - SwitchingStatus4	Variabile(*)	RWCTU-	Variabile(**)
24	Input 1B - SwitchingStatus5	Variabile(*)	RWCTU-	Variabile(**)
25	Input 1B - SwitchingStatus6	Variabile(*)	RWCTU-	Variabile(**)
26	Input 1B - SwitchingStatus7	Variabile(*)	RWCTU-	Variabile(**)
27	Input 1B - SwitchingStatus8	Variabile(*)	RWCTU-	Variabile(**)
28	Input 1B - SwitchingCommand-DedicatedStop	1 Bit	RWCTU-	[1.1] DPT_Switch
29	Input 1B - DimmingUpDownStopCommand	4 Bit	R-CT--	[3.7] [3.8] DPT_Dimming
30	Dummy - Input 1B - InfoStatus	1 Bit	-WC---	[1.1] DPT_Switch
31	Input 1B - StopStepUpDownCommand	1 Bit	R-CT--	[1.17] DPT_Trigger
32	Input 1B - MoveUpDownCommand	1 Bit	RWCTU-	[1.8] DPT_Up/Down
33	Input 1B - SceneNumberControl	1 Bytes	--CT--	[17.1] [18.1] DPT_ Scene
33	Input 1B – CounterValue 1 Byte	1 Bytes	R-CT--	[5.1] DPT_ Pulses
33	Input 1B – CounterValue 2 Byte	2 Bytes	R-CT--	[7.1] DPT _ Pulses
33	Input 1B– CounterValue 4 Byte	4 Bytes	R-CT--	[12.1] DPT_Counter pulses
34	Input 1B - Counter reset command	1 Bit	-WC-U-	[1.15] DPT_Reset
35	Input 1B - Counter runout	1 Bit	RWCTU-	[1.5] DPT_Alarm
36	Input 2A - LockCommand	1 Bit	-WC---	[1.3] DPT_Enable

37	Input 2A - SwitchingStatus1	Variabile(*)	RWCTU-	Variabile(**)
38	Input 2A - SwitchingStatus2	Variabile(*)	RWCTU-	Variabile(**)
39	Input 2A - SwitchingStatus3	Variabile(*)	RWCTU-	Variabile(**)
40	Input 2A - SwitchingStatus4	Variabile(*)	RWCTU-	Variabile(**)
41	Input 2A - SwitchingStatus5	Variabile(*)	RWCTU-	Variabile(**)
42	Input 2A - SwitchingStatus6	Variabile(*)	RWCTU-	Variabile(**)
43	Input 2A - SwitchingStatus7	Variabile(*)	RWCTU-	Variabile(**)
44	Input 2A - SwitchingStatus8	Variabile(*)	RWCTU-	Variabile(**)
45	Input 2A - SwitchingCommand-DedicatedStop	1 Bit	RWCTU-	[1.1] DPT_Switch
46	Input 2A - DimmingUpDownStopCommand	4 Bit	R-CT--	[3.7] [3.8] DPT_Dimming
47	Dummy - Input 2A - InfoStatus	1 Bit	-WC---	[1.1] DPT_Switch
48	Input 2A - StopStepUpDownCommand	1 Bit	R-CT--	[1.17] DPT_Trigger
49	Input 2A - MoveUpDownCommand	1 Bit	RWCTU-	[1.8] DPT_Up/Down
50	Input 2A - SceneNumberControl	1 Bytes	--CT--	[17.1] [18.1] DPT_ Scene
50	Input 2A – CounterValue 1 Byte	1 Bytes	R-CT--	[5.1] DPT_ Pulses
50	Input 2A – CounterValue 2 Byte	2 Bytes	R-CT--	[7.1] DPT _ Pulses
50	Input 2A – CounterValue 4 Byte	4 Bytes	R-CT--	[12.1] DPT_Counter pulses
51	Input 2A - Counter reset command	1 Bit	-WC-U-	[1.15] DPT_Reset
52	Input 2A - Counter runout	1 Bit	RWCTU-	[1.5] DPT_Alarm
53	Input 2B - LockCommand	1 Bit	-WC---	[1.3] DPT_Enable
54	Input 2B - SwitchingStatus1	Variabile(*)	RWCTU-	Variabile(**)
55	Input 2B - SwitchingStatus2	Variabile(*)	RWCTU-	Variabile(**)
56	Input 2B - SwitchingStatus3	Variabile(*)	RWCTU-	Variabile(**)
57	Input 2B - SwitchingStatus4	Variabile(*)	RWCTU-	Variabile(**)
58	Input 2B - SwitchingStatus5	Variabile(*)	RWCTU-	Variabile(**)
59	Input 2B - SwitchingStatus6	Variabile(*)	RWCTU-	Variabile(**)
60	Input 2B - SwitchingStatus7	Variabile(*)	RWCTU-	Variabile(**)
61	Input 2B - SwitchingStatus8	Variabile(*)	RWCTU-	Variabile(**)
62	Input 2B - SwitchingCommand-DedicatedStop	1 Bit	RWCTU-	[1.1] DPT_Switch
63	Input 2B - DimmingUpDownStopCommand	4 Bit	R-CT--	[3.7] [3.8] DPT_Dimming
64	Dummy - Input 2B - InfoStatus	1 Bit	-WC---	[1.1] DPT_Switch
65	Input 2B - StopStepUpDownCommand	1 Bit	R-CT--	[1.17] DPT_Trigger
66	Input 2B - MoveUpDownCommand	1 Bit	RWCTU-	[1.8] DPT_Up/Down
67	Input 2B - SceneNumberControl	1 Bytes	--CT--	[17.1] [18.1] DPT_ Scene
67	Input 2B – CounterValue 1 Byte	1 Bytes	R-CT--	[5.1] DPT_ Pulses
67	Input 2B – CounterValue 2 Byte	2 Bytes	R-CT--	[7.1] DPT _ Pulses
67	Input 2B – CounterValue 4 Byte	4 Bytes	R-CT--	[12.1] DPT_Counter pulses
68	Input 2B - Counter reset command	1 Bit	-WC-U-	[1.15] DPT_Reset
69	Input 2B - Counter runout	1 Bit	RWCTU-	[1.5] DPT_Alarm
70	Led 1 - Command	1 Bit	RWCTU-	[1.1] DPT_Switch

71	Led 2 - Command	1 Bit	RWCTU-	[1.1] DPT_Switch
72	Led 3 - Command	1 Bit	RWCTU-	[1.1] DPT_Switch
73	Led 4 - Command	1 Bit	RWCTU-	[1.1] DPT_Switch
74	Logic function 1 - Input 1	1 Bit	-WCTU-	[1.1] DPT_Switch
75	Logic function 1 - Input 2	1 Bit	-WCTU-	[1.1] DPT_Switch
76	Logic function 1 - Input 3	1 Bit	-WCTU-	[1.1] DPT_Switch
77	Logic function 1 - Input 4	1 Bit	-WCTU-	[1.1] DPT_Switch
78	Logic function 1 - Output	1 Bit	R-CT--	[1.1] DPT_Switch
79	Logic function 2 - Input 1	1 Bit	-WCTU-	[1.1] DPT_Switch
80	Logic function 2 - Input 2	1 Bit	-WCTU-	[1.1] DPT_Switch
81	Logic function 2 - Input 3	1 Bit	-WCTU-	[1.1] DPT_Switch
82	Logic function 2 - Input 4	1 Bit	-WCTU-	[1.1] DPT_Switch
83	Logic function 2 - Output	1 Bit	R-CT--	[1.1] DPT_Switch
84	Logic function 3 - Input 1	1 Bit	-WCTU-	[1.1] DPT_Switch
85	Logic function 3 - Input 2	1 Bit	-WCTU-	[1.1] DPT_Switch
86	Logic function 3 - Input 3	1 Bit	-WCTU-	[1.1] DPT_Switch
87	Logic function 3 - Input 4	1 Bit	-WCTU-	[1.1] DPT_Switch
88	Logic function 3 - Output	1 Bit	R-CT--	[1.1] DPT_Switch
89	Logic function 4 - Input 1	1 Bit	-WCTU-	[1.1] DPT_Switch
90	Logic function 4 - Input 2	1 Bit	-WCTU-	[1.1] DPT_Switch
91	Logic function 4 - Input 3	1 Bit	-WCTU-	[1.1] DPT_Switch
92	Logic function 4 - Input 4	1 Bit	-WCTU-	[1.1] DPT_Switch
93	Logic function 4 - Output	1 Bit	R-CT--	[1.1] DPT_Switch

Table A1. Dimensions and DPT for Communication Objects with Independent Inputs:

Dimens. (*)	***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 percentua	[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 float value	[9.*] 2-byte float value

10.2 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:

SBS S.p.A. Via Circonvallazione s / n, I-28010 Miasino (NO) Italy.

10.3 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
- ekinex® is a registered trademark of SBS S.p.A.
- KNX® and ETS® are registered trademarks of KNX Association cvba, Brussels