Capacitive Touch Switches

User Manual Version: [1.7]_a

www.zennio.com

USER MANUAL

CONTENTS

1 INTRODUCTION

The **multifunction capacitive touch switch** from Zennio are a fully customisable solution for the control of rooms where user control of air conditioning systems, lighting, blinds, scenes, etc. is required.

They are offered at **a reduced size** and **weight**, with **one, two, four, six, eight or ten** capacitive touch buttons (according to the user's needs) with LED backlight to confirm the press of the buttons as well as showing states.

Important: *Please note that the functionalities described in this document and their ETS configuration may be slightly different or not be included depending on the device. For detailed information, please refer to the user* ***manual of the specific device****, at the Zennio website (www.zennio.com).*

2 CONFIGURATION

Please note that the screenshots and object names shown next may be slightly different depending on the device and on the application program.

2.1 GENERAL

In order to allow the device to perform the desired functions, a number of options must be parameterized, either related to its **general behaviour** (horizontal/vertical orientation, sounds, LED brightness levels…) or to **advanced features** (lock procedure of the touch panel, cleaning function, welcome back object).

2.1.1 CONFIGURATION

In the "Configuration" tab, the general settings are displayed. Most are checkboxes that enable/disable other functionalities

ETS PARAMETERISATION

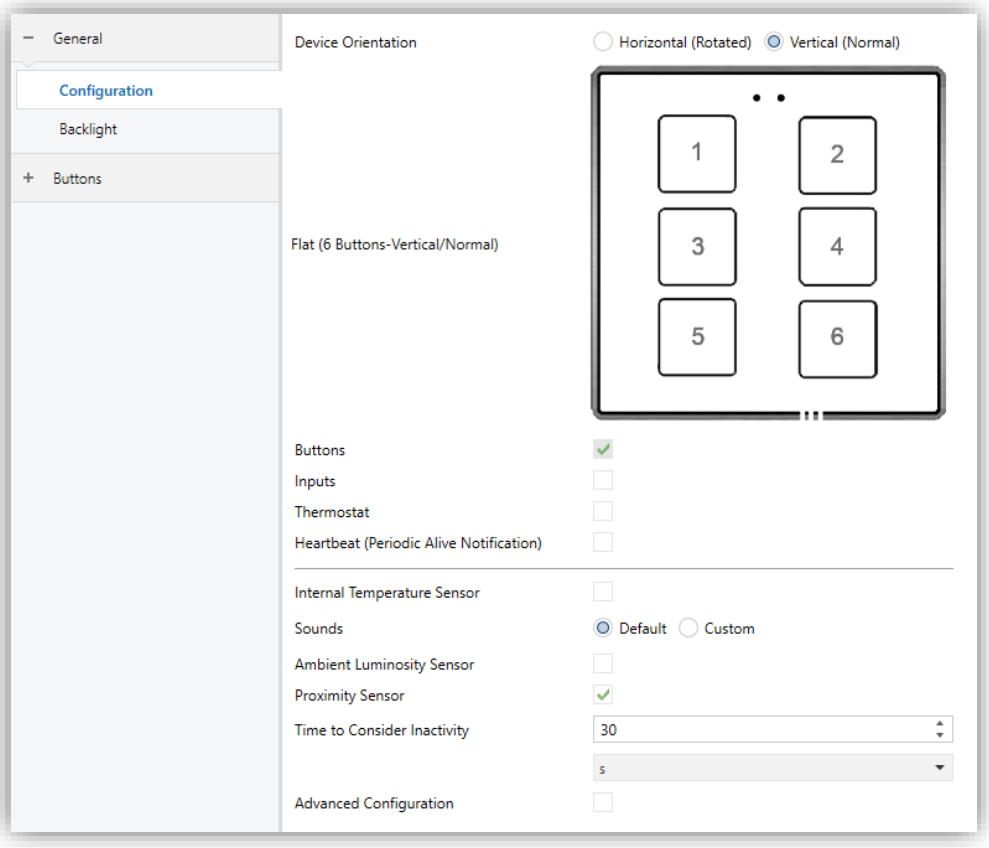

Figure 1. Main configuration.

With regard to the configuration of the capacitive push buttons, the following parameters are available (the rest will depend on the device):

- **Buttons** [*enabled*][1]: read-only parameter to make it evident that the "Buttons" tab is always enabled in the tab tree on the left. See section 2.2 for details.
- **Sounds** [*Default / Custom*]: sets whether the sound functions (button beeps, alarm and doorbell) should work according to the pre-defined configuration or to a user-defined configuration. See section 2.1.2 for details.
- **Advanced Configuration** [*disabled/enabled*]: enables or disables the "Advanced" tab in the tree on the left. See section 2.1.3 for details.

For more detailed information about the rest of the parameters, please refer to the user **manual of the specific device**, at the Zennio website *(www.zennio.com)*.

2.1.2 SOUNDS

Capacitive touch switches are able to emit a brief beep as an **acoustic feedback when a button is pressed**.

Enabling the button sounds can be done either by parameters or through an object, being also possible to define in parameters if the button sounds should be initially enabled or not.

Moreover, these can also emit the following sounds on request (through the corresponding communication objects) if enabled:

- **Doorbell sounds**: a single beep.
- **Alarm sounds**: a sequence of brief beeps with a higher pitch. The sequence will only stop when the alarm object gets deactivated or when the user touches any of the buttons (this, moreover the deactivation alarm, will trigger the button action).

The range of sounds emitted will be different depending on the sound type selected.

[1] The default values of each parameter will be highlighted in blue in this document, as follows: [*default*/rest of options].

ETS PARAMETERISATION

In case the default button beep sound matches the requirements of the installation and the doorbell and alarm functions are not necessary, the **Sounds** parameter in the general "Configuration" tab (see section 2.1.1) can be set to "Default". This will also imply that the button beeps will be unconditional, as it will not be possible to disable this function through an object.

On the other hand, if set to "Custom", a specific tab named "Sounds" will show up in the tab tree on the left. The initial configuration of this screen is equivalent to the aforementioned default option. However, the following parameters will be configurable.

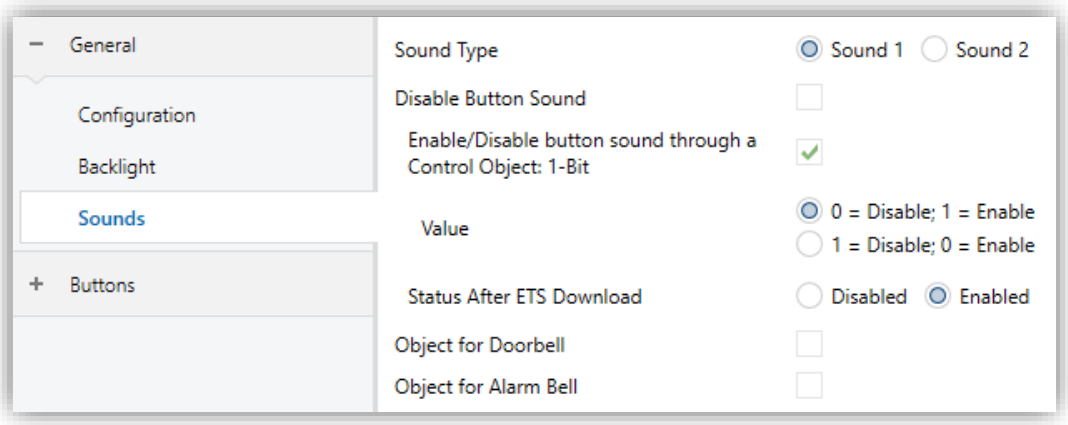

Figure 2. Sounds

The default configuration of this tab is equivalent to the one mentioned above. However, the following parameters can be customized:

- **Sound Type** [*Sound 1 / Sound 2*]: sets which sounds range incorporates the device.

- **Disable button sound** [*disabled/enabled*]: enables the buttons beeping. If enabled, the following parameters will also be available:

  - **Enable / Disable button sounds through a 1-bit object** [*disabled/enabled*]: makes it possible to disable / resume the button beeping function in runtime by writing to a specific object ("**[General] Sounds – Disabling button sound**"). If enabled, it will be shown:

    - **Value** [*0 = Disabled; 1 = Enabled / 1 = Disabled; 0 = Enabled*]: configures the values that will disable/enable the acoustic signals after pressing.

- **Status After ETS Download** [*enabled*/*disabled*]: sets whether the button beeping function should start up enabled or disabled after an ETS download.

- **Object for Doorbell** [*disabled*/*enabled*]: enables or disables the doorbell function. If enabled, a specific object ("**[General] Sounds - Doorbell**") will be included into the project topology.

- **Object for Alarm Bell** [*disabled*/*enabled*]: enables or disables the alarm function. If enabled, a specific object ("**[General] Sounds - Alarm**") will be included into the project topology.

2.1.3 ADVANCED

Independent tab for the parameterisation of some advanced functions. These functions are explained next.

ETS PARAMETERISATION

After enabling the **Advanced configuration** from "Configuration" screen (see section 2.1.1), a new tab will be incorporated into the tree on the left.

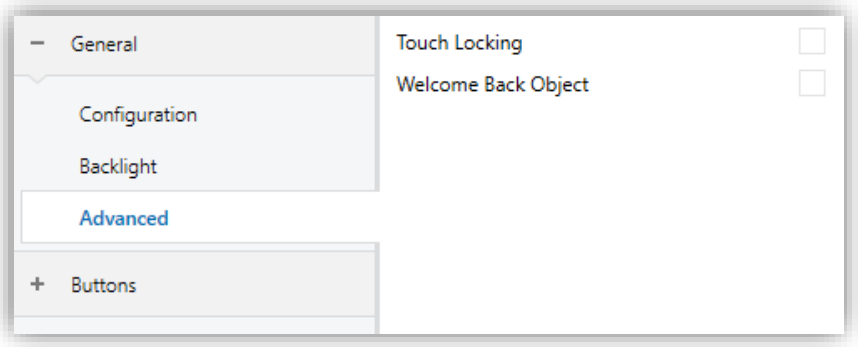

Figure 3. Advanced

- **Touch locking** [*disabled*/*enabled*]: enables or disables the "Touch locking" tab in the tree on the left. See section 2.1.3.1 for details.

- **Welcome back object** [*disabled*/*enabled*]: enables or disables the "Welcome back" tab in the tree on the left. See section 2.1.3.2 for details.

2.1.3.1 TOUCH LOCKING

The touch panel of capacitive touch switches can be optionally locked and unlocked anytime by writing a configurable one-bit value to a specific object provided for this purpose. It can also be done through scene values.

While locked, pressing on the buttons will be ignored: no actions will be performed (and no LEDs will change their states) when the user touches on any of the controls.

ETS PARAMETERISATION

After enabling **Touch Locking** in "Advanced" tab, a new tab will be incorporated into the tree on the left.

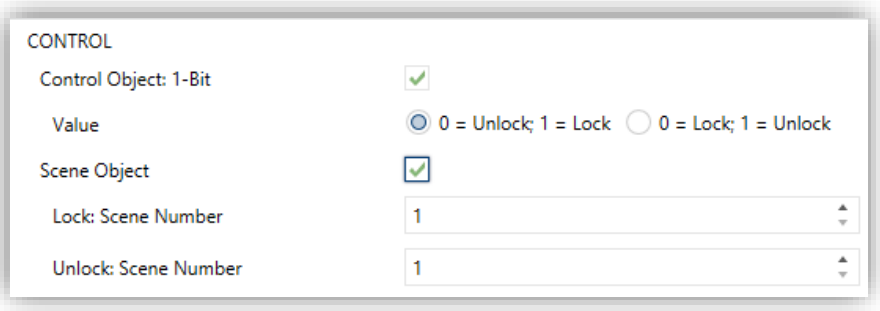

Figure 4. Touch Locking; Control

- **Control Object: 1-Bit** [*disabled*/*enabled*]: when marked, a specific drop-down list will show up to select which value should trigger which action.
  - **Value** [*0 = Unlock; 1 = Lock* / *0 = Lock; 1 = Unlock*]: these values are received through the object "**[General] Touch Locking**".
- **Scene Object** [*disabled*/*enabled*]: when marked, two specific textboxes will show up to enter the scene numbers (1 - 64) that should trigger each action. These values are to be received through the general "**[General] Scene: Receive**" object.

2.1.3.2 WELCOME BACK OBJECT

Capacitive touch switches can send a specific object (the *welcome back object*) to the KNX bus when the user presses a touch button after a significant amount of time since the last or presence detection. Sending it or not can also depend on an **additional, configurable condition** consisting in the evaluation of up to five binary objects.

Any actions that in normal operation may be executed will not be if the welcome back object is sent to the bus. Thus, if the user presses a button and this causes that the welcome back object is sent, the normal action of that button will not be triggered.

On the other hand, if the additional condition is not evaluated to true, the device will react normally. Hence, the action corresponding to the button touch will be executed.

The welcome back object can consist in a **one-bit** value or a **scene** value (or both), depending on the parameterisation.

ETS PARAMETERISATION

After enabling **Welcome Back Object**, a new tab will be incorporated into the tree on the left.

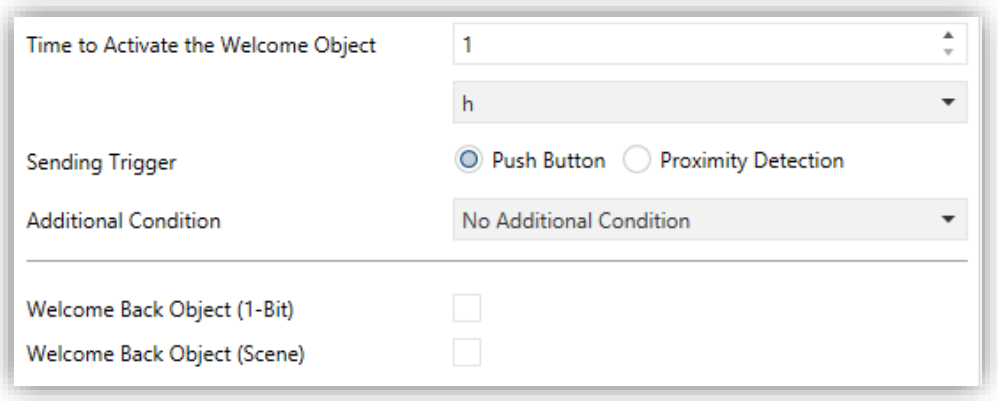

Figure 5. Welcome Back Object.

This screen contains the following parameters:

- **Timeout to Activate the Welcome Object** [*1…255*] [*s/min/h*]: sets the minimum time that should elapse after the last button touch (or presence detection, when the proximity sensor is enabled) before the next one triggers the execution of the welcome back function.

- **Sending Trigger** [*Push Button / Proximity Detection*]: sets whether the welcome back object is sending after a touch in the screen or when the proximity sensor detects presence.

- **Additional Condition**: sets if sending the welcome back object should also depend on an external condition. The option by default is [*No Additional Condition*]. The following are available too:

- [*Do not send unless all additional conditions are 0*]: the welcome back object will only be sent if all the condition objects are found to have the value “0”.
- [*Do not send unless all additional conditions are 1*]: the welcome back object will only be sent if all the condition objects are found to have the value “1”.
- [*Do not send unless at least one of the additional conditions is 0*]: the welcome back object will only be sent if at least one of the condition objects is found to have the value “0”.
- [*Do not send unless at least one of the additional conditions is 1*]: the welcome back object will only be sent if at least one of the condition objects is found to have the value “1”.

- **Welcome Back Object (1-Bit)** [*disabled*/*enabled*]: checkbox to enable the sending of a 1-bit value (through “**[General] Welcome back**”) when the welcome back function is triggered and the condition (if any) evaluates to true. The desired value should set in **Value** [*Send 0* / *Send 1*].

- **Welcome Back Object (Scene)** [*disabled*/*enabled*]: checkbox to enable the sending of a scene run request (through “**[General] Scene: send**”) when the welcome back function is triggered and the condition (if any) evaluates to true. The desired value should be set in **Scene Number** [*1…64*].

2.2 BUTTONS

Capacitive touch switches have **one, two, four, six, eight or ten buttons** at the user's disposal for the execution of actions.

The distribution of the buttons will depend on the device chosen and the orientation selected in "Configuration" (see section 2.1.1), being possible to configure them as single-button controls or in pairs by **combining any two of them**.

- **Capacitive Touch Switches with only one button**: only one individual control is possible (two-button controls are not available). Moreover, it can only be configured under the normal (vertical) orientation.
- **Capacitive Touch Switches with two buttons**: up to two individual controls or a pair.
- **Capacitive Touch Switches with four buttons:** up to four one-button controls can be configured, or up to two two-button controls.
- **Capacitive Touch Switches with six buttons**: up to six one-button controls, or three two-button controls can be configured.
- **Capacitive Touch Switches with eight buttons**: up to eight one-button controls, or four two-button controls can be configured.
- **Capacitive Touch Switches with six buttons**: up to ten one-button controls, or five two-button controls can be configured.

2.2.1 CONFIGURATION

The following is a list of the functions that can be assigned to each button.

- **Disabled** (the button will not react to user presses)**.**
- **Pair A**, **B, C, D or E** (the number of available pairs depends on the selected model), being the function of such pair one of the following:
  - Switch (binary).
  - Two objects (short press / long press).
  - Light dimmer.
  - Shutter.

- **Individual** (one-button control):
  - Switch (Binary).
  - Hold & release.
  - Two objects (short press / long press).
  - Scene.
  - Scaling constant.
  - Counter constant.
  - Float constant.
  - Light dimmer.
  - Shutter.
  - LED indicator.
  - Room State

Apart from the button function itself, the desired behaviour of the button LEDs can be set. The different illumination modes have been detailed in ANNEX I. LED Illumination Modes.

The next sections explain the configuration involved for each of the above functions.

ETS PARAMETERISATION

An independent tab for the parameterisation of the buttons is shown in ETS by default, initially containing only a sub-tab named “Configuration”.

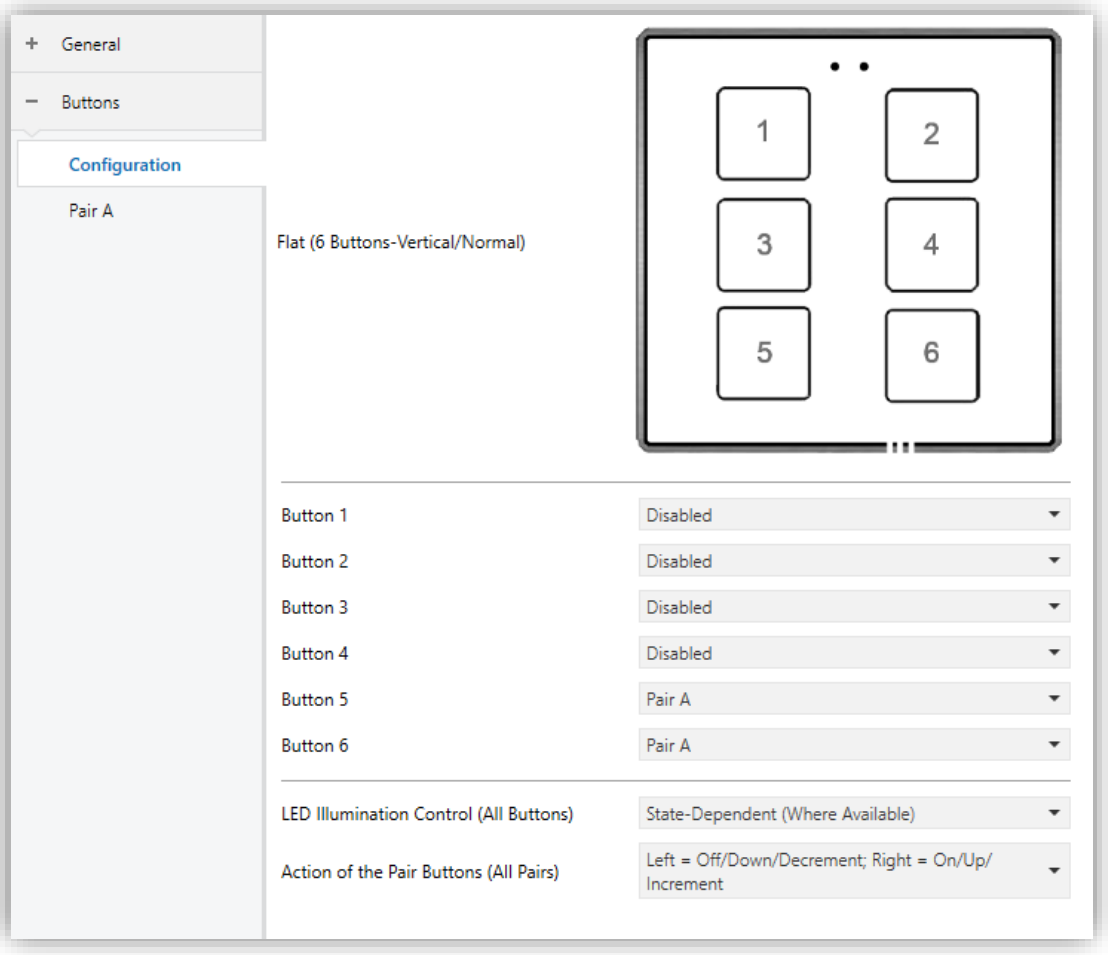

Figure 6. Buttons - Configuration

One drop-down list with the following options is shown per **button**:

- [*Disabled*]. See section 2.2.2 for details.
- [*Individual*]. Selecting this option brings a new tab ("Button Ix", where "x" depends on the button), which will make it possible to configure the functionality of that particular touch button. See section 2.2.3 for details.
- [*Pair X*]. Sets that this touch button will belong to a two-button control (where X is A, B, C, D or E, depending on the model). Once one pair has been assigned to two buttons (and not before), a new tab ("Pair X") will show up in the tab tree, in order to configure the desired functionality. See section 2.2.4 for details.

A drop-down list (**LED Illumination Control (All buttons)**) is provided at the bottom of the window so a joint behaviour can be configured for the illumination of the LEDs. The options are (please refer to ANNEX I. LED Illumination Modes for details):

- [*Regular*]
- [*State-Dependent (where available)*]
- [*State-Dependent (where available) (both LEDs)*]
- [*Dedicated Object*]
- [*Configure every button (pair) separately*]: in case of selecting the last option, there will be a specific parameter **for each control** to specifically select the desired behaviour of the LED (or LEDs).

Finally, if at least one two-button control is being configured (either Pair X), an additional parameter (**Action of the pair buttons (all pairs)**) will be available to determine an operation criterion. The options are:

- [*Left = Off/Down/Decrement; Right = On/Up Increment*]
- [*Right = Off/Down/Decrement; Left = On/Up Increment*]
- [*Every button pair is configured separately*]

2.2.2 DISABLED

While a button stays disabled, it will not be functional: touching on it will not cause the execution of actions, nor will make the associated LED light.

ETS PARAMETERISATION

This function has no related parameters.

2.2.3 INDIVIDUAL

Buttons configured to work as individual (separate) controls can be assigned any of the following control functions:

- **LED indicator**: user presses will not trigger any function although the LED will turn on or off depending on the values received from the bus.

- **Switch**: whenever the user touches the button, a binary value will be sent to the KNX bus. This value is configurable and may be 0 or 1, or alternate with every touch according to the sequence 1 → 0 → 1 → …

  Under a "State-dependent" LED illumination, the LED will remain on/off according to the current state (on/off) of the object.

- **Hold & Release**: as soon as the user touches the button, a binary value ("0" or "1", configurable) will be sent to the KNX bus. And as long as the user releases the button, another value ("0" or "1", also configurable) will be sent through the same object.

  The "State-dependent" LED mode is not available for this function.

- **Two Objects (Short Press / Long Press)**: specific binary values will be sent both after a short or a long press (a different object will be used in each case). Under a "State-dependent" LED illumination, the LED will remain on/off according to the current state (on/off) of either one object or the other, which can be configured in parameters. However, if **LED Illumination Control** (**All Buttons)** has been set to "State-dependent (where available)", only the short press object will apply.

- **Scene**: after the user touches the button, an order to run a specific scene (configurable) will be sent to the bus. If enabled in parameters, orders to save

the scene can also be sent to the bus after a three-second press on the button. The "State-dependent" LED mode is not available for this function.

- **Scaling Constant**: sends a percentage value (configurable) to the bus when the user touches the button. Under a "State-dependent" LED illumination, the LED will remain on/off depending on whether the current value of the object matches the one parameterised. This object can also be written from the bus, which will update the LED according to the new value.

- **Counter Constant**: sends an integer value (configurable) to the bus when the user touches the button. This value can be one-byte or two-byte sized, as well as signed or unsigned. The available ranges are shown next:

	1-byte	2-byte
Unsigned	0 – 255.	0 – 65535.
Signed	-128 – 127.	-32768 – 32768.

Table 1 Value range – Counter type constant

The "State-dependent" LED illumination mode is analogous as for the Scaling Constant function.

- **Float Constant**: sends a two-byte floating point value (configurable) to the bus when the user touches the button. The available range is -671088.64 to 670433.28.

  The "State-dependent" LED illumination mode is analogous as for the Scaling Constant and Counter Constant functions.

- **Dimmer**: implements a one-button light control that sends orders to the KNX bus, which can then be executed by light dimmers. These orders consist in:

  - Switch-on/Switch-off orders (on short presses).
  - Step dimming orders (on long presses) and the subsequent stop order once the button is released.

  Being a one-button control, the **switch orders will alternate** (on/off) for every short press, and so will do the step dimming orders (increase/decrease) for every long press. However, there are some exceptions:

- On a long press: an increase dimming order will be sent if the light is found to be off (according to the status object). On the other hand, a decrease order will be sent if it is found to be 100%.

- On a short press: a switch-on order will be sent if the light is found to be off (according to the status object). On the other hand, a switch-off order will be sent if it is found to be on (value greater than 0%).

Note that the device considers that the **current light level** is the value of a specific one-byte object provided to be written from the KNX bus (i.e., to receive feedback from the dimmer). This object is internally updated after a short or long press, but linking it to the real dimmer status is highly advisable.

Under a "state-dependent" LED illumination, the LED will remain on/off according to the value of the aforementioned status object (i.e., off when the value is 0% and on in any other case).

Note: *after a bus recovery, the light dimmer should send back the status object so the control and the LED update their own state, instead of simply recovering the previous one.*

- **Shutter**: implements a one-button shutter control that sends orders to the KNX bus, which can then be executed by an actuator.

Two control types can be configured:

- Standard: the device will react to both long and short presses, being possible to send the bus the following commands:

  - Move (raise/lower) orders (on **long presses**).
  - Stop/Step orders (on **short presses**).

Being a one-button control, the direction of the motion will alternate (upwards/downwards) for both the move and the step orders after every long press. However, there are some exceptions to this alternation:

- On a short press: a step-up order will be sent if the last long press made the shutter move up, or if the current position is found to be 100%. On the other hand, a step-down order will be sent if the last long press made the shutter move down or if the current position is found to be 0%.

- On a long press: a move-up order will be sent if the last short press caused a step-down order or if the current position is found to be 100%. On the other hand, a move-down order will be sent if the last short press caused a step-up order or if the current position is found to be 0%.

As usual in the KNX standard, **stop/step** orders are interpreted by the actuators as a request to move the slats one step up or down (in case the shutter is still) or as a request to interrupt the motion of the shutter (in case it is already moving up or down).

Capacitive touch switches are aware of the **current position of the shutter** through a specific object which should be linked to the analogous object of the shutter actuator in order to receive feedback. This object is initialised with value "50%" after a download or a bus failure; therefore, the actuator is required to update it with the real value after the bus recovery.

➢ Hold & Release: the device will send an order to move the shutter when the button is touched, and the order to stop it as soon as it is released. Hence, short or long touches have the same effect: the shutter will remain in motion as long as the user keeps holding the button.

The direction of this motion (upwards or downwards) will **alternate** with every touch, according to the following sequence: downwards → upwards → downwards → …

However, there are some exceptions to this alternation:

- If the position of the shutter is found to be 0%, the next order will lower the shutter.
- If the position of the shutter is found to be 100%, the next order will raise the shutter.

The "state-dependent" LED illumination mode is not available for this function.

- **Room State:** allows controlling the room states (normal, make up request, do not disturb). Pressing the button will activate the *Do Not Disturb* or *Make-Up Request* status (as configured) or deactivate it to return to *Normal* status.

Depending on the parameterisation and the current value of object, after a short press the following values will be transmitted.

Parameterisation	Current Object Value	Transmitted Value
Make-Up Request	Do Not Disturb / Normal	Make-Up Request
	Make-Up Request	Normal
Do Not Disturb	Normal / Make-Up Request	Do Not Disturb
	Do Not Disturb	Normal

Table 2 Room States

If the LED illumination es "State-dependent", LED will light up when the current object value coincides with the parameterized value.

ETS PARAMETERISATION

When an individual button has been enabled, a specific tab ("Button In") becomes available under "Buttons" in the tree on the left.

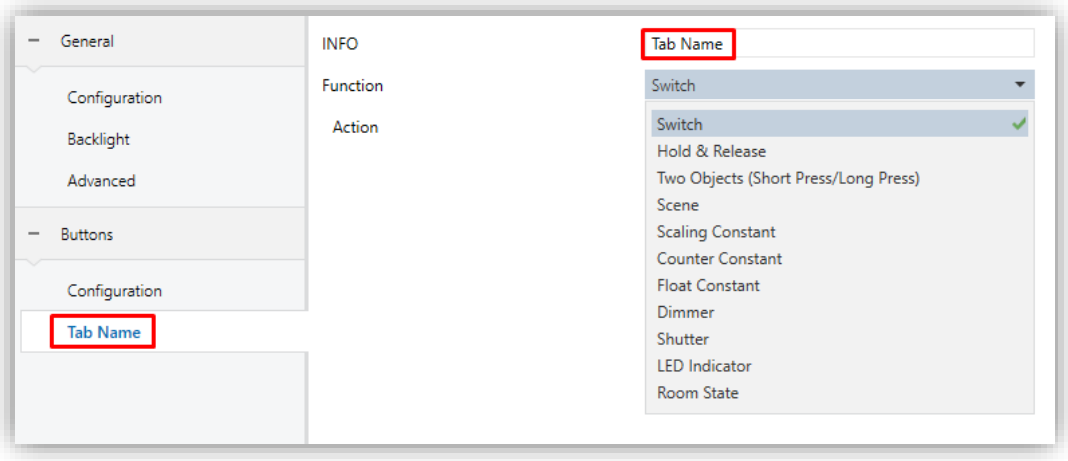

Figure 7. Individual Button.

Textbox **INFO** allows changing the default name of the tab in the left menu, as shows the following figure.

The main parameter that needs to be configured is:

- **Function** [*Switch / Hold & Release / Two Objects (short press / long press) / Scene / Scaling constant / Counter constant / Float constant / Dimmer / Shutter / LED indicator / Room State*]: sets the desired function for the button.

In case the option "Configure every button (pair) separately" has been selected in the **LED Illumination Control Parameter (All Buttons)** in the "Configuration" tab (see section 2.2.1), the additional parameter will be displayed:

- **LED Illumination Control** [*State-Dependent / Regular / Dedicated Object*].

In case of selecting the latter, the object "**[Btn] [PX] Led On/Off**" will be included in the project topology and a new parameter to select the **value** [*0 = Off; 1 = On / 0 = On; 1 = Off*] to switch off and on the LED shows up:

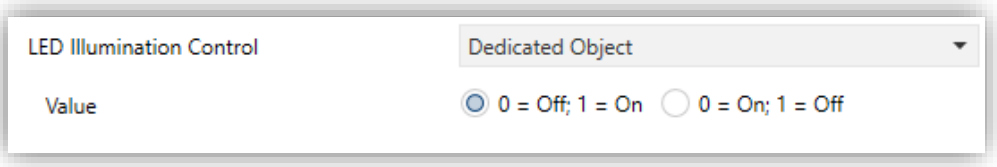

Figure 8. Button Pair – LED Illumination – Dedicated Object

Note: *For further information, please refer to section 2.2.1 and ANNEX I. LED Illumination Modes.*

Depending on the function, some more parameters are involved (as described next). Please note that in the next pages "**[In]**" is used as a general notation for the communication objects, where "n" depends on the particular button pair.

LED Indicator

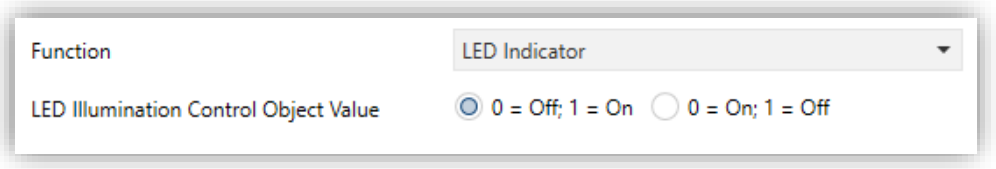

Figure 9. Individual Button – LED indicator.

- **LED illumination control object value** [*0 = Off; 1 = On / 0 = On; 1 = Off*]: sets the behaviour of the LED of the button. The options are similar to those of the dedicated-object LED illumination available for other control types.

  Note: *this parameter does not depend on the option selected for **LED Illumination Control (All Buttons)** (see section 2.2.1).*

After assigning this function to the button, object "**[Btn] [In] LED On/Off**" is included in the project topology, so that the values that determine the state of the LED at a given time can be received from the bus.

Switch

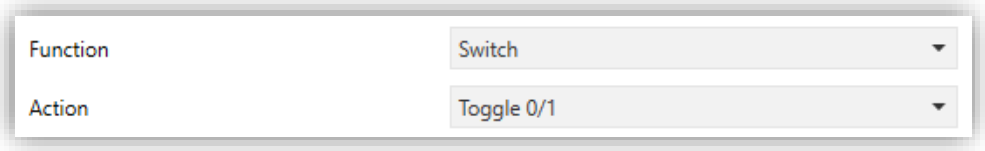

Figure 10. Individual Button - Switch.

- **Action** [*Toggle 0/1 / Send 0 / Send 1*]: sets the value to be sent to the bus (through object "**[Btn] [In] Switch**") when the user touches the button.

Hold & Release

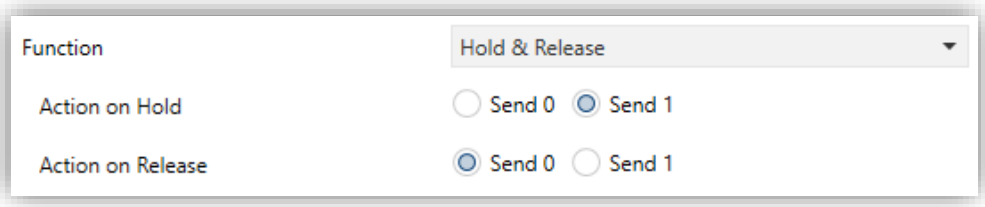

Figure 11. Individual Button - Hold & Release.

- **Action on Hold** [*Send 1 / Send 0*]: sets the value to be sent to the bus (through "**[Btn] [In] Hold & Release**") when the user touches the button.
- **Action on Release** [*Send 0 / Send 1*]: sets the value to be sent to the bus (again, through "**[Btn] [In] Hold & Release**") when the user stops touching the button.

Two Objects (Short Press / Long Press)

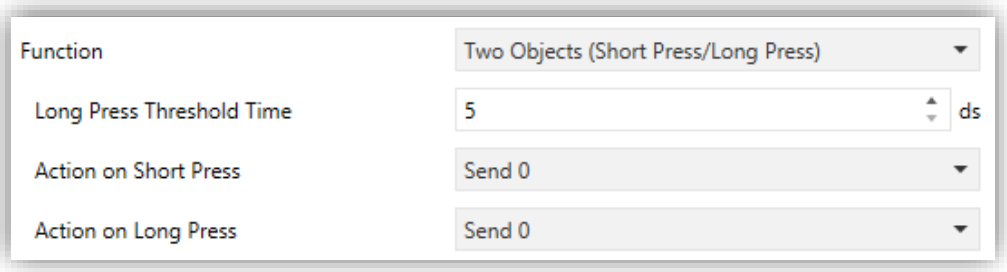

Figure 12. Individual Button - Two Objects (Short Press / Long Press).

- **Long Press Threshold Time** [*0…5…50 ds*]: sets the minimum time the user should hold the button in order to consider it a long press.

- **Action on Short press** [*Send 0 / Send 1 / Toggle 0/1 / Send 1-byte value*]: sets the value to be sent to the bus (through "**[Btn] [In] Two switches - Short press**") when the user short-presses the button.

  In case of selecting the latter, an additional parameter (**Value** [*0...255*]) will be displayed to enter the desired one-byte value.

- **Action on Long press** [*Send 0 / Send 1 / Toggle 0/1 / Send 1-byte value*]: sets the value to be sent to the bus (through "**[Btn] [In] Two switches - Long press**") when the user long-presses the button.

Scene

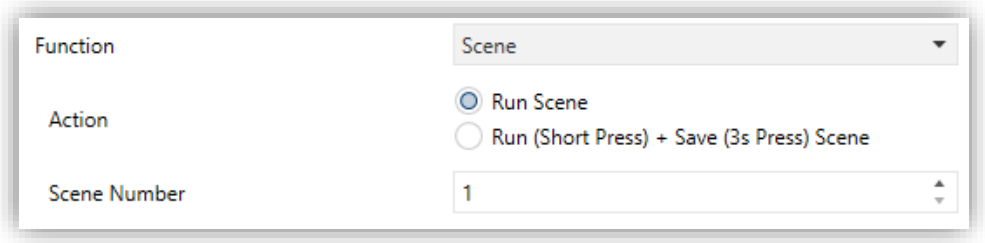

Figure 13. Individual Button - Scene.

- **Action** [*Run scene / Run (short press) + Save (3s press) scene*]: sets whether the value to be sent to the KNX bus (through "**[General] Scene: send**") when the user touches the button will always be a scene run request or –depending on the length of button press– a scene run or save request.

- **Scene number** [*1…64*]: number of the scene to be sent to the bus, both in the case of the run requests and the save requests.

Scaling Constant / Counter Constant / Float Constant

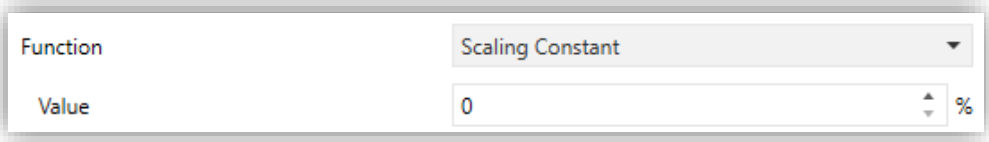

Figure 14. Individual Button - Scaling Constant

- **Value** [*0*]: sets the value to be sent to the KNX bus when the user touches the button. The available range and the object through which the value is sent depend for each case, as the table below shows.

In case of selecting Counter Constant, two specific parameters (**Size** and **Signed**) will be displayed to respectively define the size of the constant ("1 byte" or "2 bytes") and whether it is a signed value or an unsigned value. Depending on that, the range and the name of the object will vary.

	Available Values	Name of the Object
Scaling Constant	0% – 100%	[Btn] [In] Scaling
Counter Constant	0 – 255 -128 – 127 0 – 65535 -32768 – 32767	[Btn] [In] Counter – 1-Byte unsigned [Btn] [In] Counter – 1-Byte Signed [Btn] [In] Counter – 2-Byte Unsigned [Btn] [In] Counter – 2-Byte Signed
Float Constant	-671088.64 – 670433.28	[Btn] [In] Float

Table 3 Constant type numerical control

Dimmer

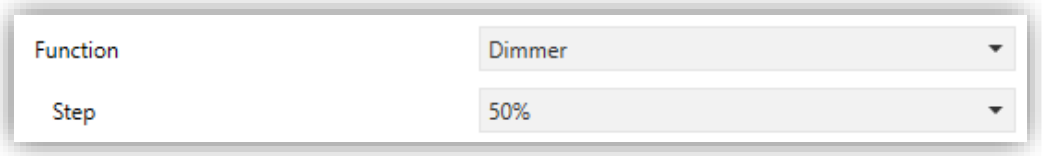

Figure 15. Individual Button - Dimmer

The (alternating) switch orders will be sent through the "**[Btn] [In] Light - On/Off**" one-bit object, while the (alternating) increase/decrease/stop orders will be through the "**[Btn] [In] Light - Dimming**" four-bit object.

On the other hand, the "**[Btn] [In] Light - Dimming (Status)**" one-byte object may be linked to the light level status object of the dimmer (in fact, this object is only intended to receive values from the bus, not to send them). As explained at the beginning of this section, the state-dependent LED lighting will be determined by the value of this object (LED off at 0% and on at any other level).

The parameters for this function are:

- **Step** [*100% / 50% / 25% / 12,5% / 6,25% / 3,1% / 1,5%*]: defines the dimming step to be sent (through "**[Btn] [In] Light - Dimming**") to the light dimmer with every long press.

  Note: *since dimmers typically do not apply the new light level immediately (i.e., the step is performed progressively) and since capacitive touch switches send*

an order to interrupt the step dimming once the user releases the button, it is advisable to configure a step of 100%.

This way, the user can perform any dimming step by simply leaving the button pressed and then releasing it, without needing to make successive button presses.

Shutter

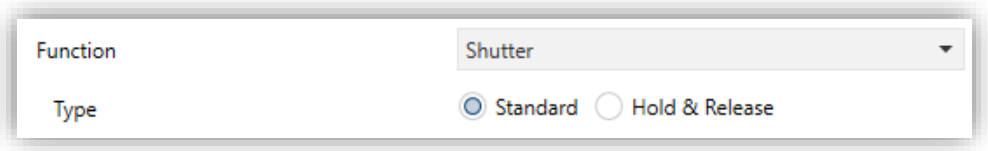

Figure 16. Individual Button - Shutter

The (alternating) move up/down orders will be sent through the "**[Btn] [In] Shutter - Move**" one-bit object, while the (alternating) step up/down orders will be through the "**[Btn] [In] Shutter - Stop / Step**" one-bit object.

Additionally, a one-byte object ("**[Btn] [In] Shutter Position**") is provided to link it to the position status object of the shutter actuator (in fact, this object is only intended to receive values from the bus, not to send them).

The parameters for this function are:

- **Type** [*Standard / Hold & Release*]: sets the desired control type.

Room State

When this function is assigned to the button, the object for the control ""**[Btn][In] Room State**" is enabled. This object will also be a status indicator.

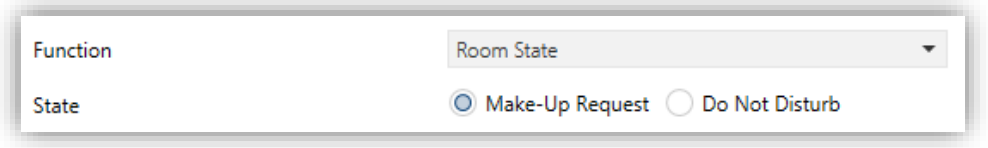

Figure 17. Individual Button – Room state

- **State** [*Make-up Request / Do not Disturb*]: sets the state that is activated with this button. Commutes between Normal ("0") and the selected state: Make-up room ("1") and Do not disturb ("2").

2.2.4 PAIR

Buttons configured to work as a joint control can be assigned the following functions:

- **Switch**: pressing one of the two buttons will make capacitive touch switches send a binary value to the bus, while pressing on the other will make it send the inverse binary value. It is possible to configure which one does what.

  Under a "state-dependent" LED illumination (see ANNEX I. LED Illumination Modes), the LED of the corresponding button will remain on/off according to the current state (on/off) of the switch. On the other hand, under a "state-dependent (both LEDs)" LED illumination, both of them will remain on while the switch is in the "on" state, and off while in the "off" state.

- **Two Objects (Short Press / Long Press)**: permits sending specific binary values both after a short or a long press on any of the two buttons (i.e., they will work as a joint control; for independent buttons, please configure them as individual). Different objects will be used for the short and long presses.

  Moreover, it is possible (in parameters) to make the "state-dependent" and "state-dependent (both LEDs)" LED illumination modes (see ANNEX I. LED Illumination Modes) depend on either one object or the other. However, if **LED Illumination Control (All Buttons)** has been set to "state-dependent (where available)" only the short press object will be considered

- **Dimmer**: short-pressing one of the two buttons will make capacitive touch switches send a switch-on order to the bus, while doing so on the other button will make it sends a switch-off order.

  Long presses will make it send a step dimming order (the value of which is configurable) to make a dimmer increase or decrease the light level (and a stop order as soon as the user releases the push button). It is possible to configure which button does what.

  Under a "state-dependent" LED illumination (see ANNEX I. LED Illumination Modes), the LED of the corresponding button will remain on/off according to whether the current value of the light level status object (which should be updated by the actual dimmer) is greater than 0% or not. On the other hand, under a "state-dependent (both LEDs)" LED illumination, both together will remain on or off depending on such value.

- **Shutter**: this option permits making use of the two buttons to control a shutter actuator connected to the bus. Two alternative control methods are possible:

  - Standard: a long press will make the device send to the KNX bus an order to start moving the shutter (upwards or downwards, depending on the button), while a short press will make it send a stop order (which will be interpreted as an order to step up or to step down –depending on the button– if the shutter was not in motion and such function is available).

  - Hold & Release: as soon as the button is held, the device will send the KNX bus an order to start moving the shutter (upwards or downwards, depending on the button). Once the button is released, it will send an order to stop the shutter.

  The "state-dependent" and "state-dependent (both LEDs)" LED illumination modes are not available for this function (only the "regular" and "dedicated object" LED illumination are available). See ANNEX I. LED Illumination Modes for details.

ETS PARAMETERISATION

Once two buttons have been assigned to a particular pair, a specific tab ("Pair X") becomes available under "Buttons" in the tab tree.

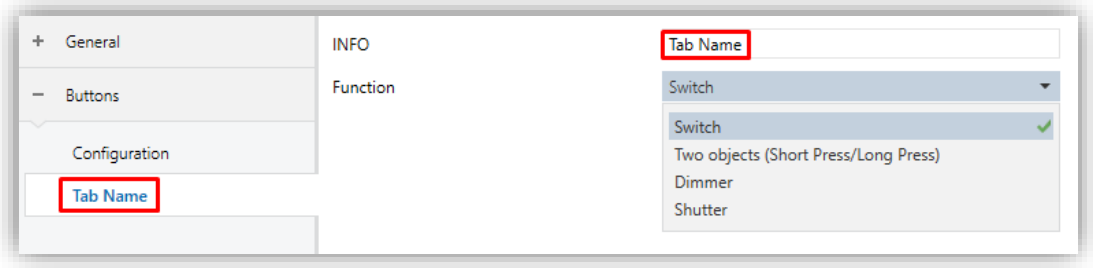

Figure 18. Button Pair.

Textbox **INFO** allows changing the default name of the tab in the left menu, as shows the following figure.

The main parameter that needs to be configured is:

- **Function** [*Switch / Two objects (short press / long press) / Dimmer / Shutter*]: sets the desired function for the button pair.

In case the option "Configure every button (pair) separately" has been selected in the **LED Illumination Control Parameter (All Buttons)** in the "Configuration" tab (see section 2.2.1), the additional parameter will be displayed:

- **LED Illumination Control** [*State-Dependent / State-dependent (both LEDs) / Regular / Dedicated Object*].

  In case of selecting the latter, the object "**[Btn] [PX] Led On/Off**" will be included in the project topology and a new parameter to select the **value** [*0 = Off; 1 = On / 0 = On; 1 = Off*] to switch off and on the LED shows up:

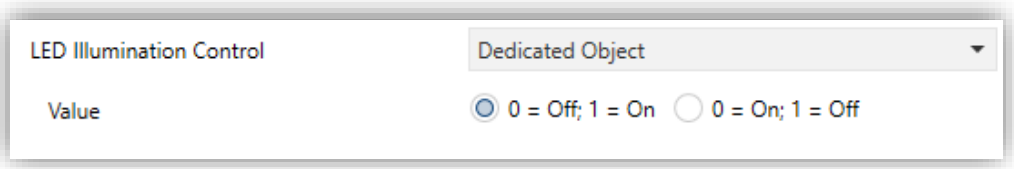

Figure 19. Button Pair – LED Illumination – Dedicated Object

Note: *For further information, please refer to section 2.2.1 and ANNEX I. LED Illumination Modes.*

Depending on the function, some more parameters are shown, as described next. Please note that in the next pages the general notation "**[X]**" is used for the name of the communication objects, as "X" depends on the button pair (A, B or C).

Switch

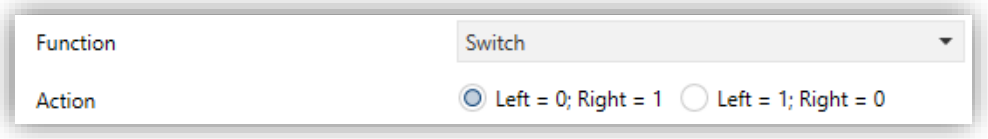

Figure 20. Pair Buttons – Switch.

- **Action** [*Left=0; Right=1/Left=1; Right=0*]: assigns each of the two buttons the value to be sent through "**[Btn] [PX] Switch**" (which has the Write flag enabled, so the state of the switch can be updated from external devices).

  Note: *this parameter will remain hidden unless having selected "Every button pair is configured separately" in* ***Action of the pair buttons****.*

Two Objects (short Press / long press)

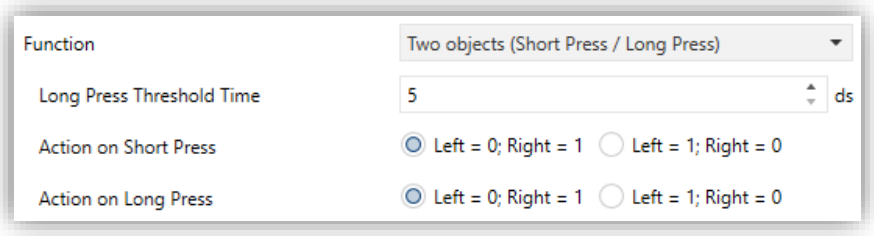

Figure 21. Pair Buttons - Two Objects (Short Press / Long Press).

- **Long Press Threshold Time** [*5…50*][*ds*]: sets the minimum time the user should hold the button in order to consider it a long press. The available range is 5 to 50 tenths of a second, being 5 tenths the default value.

- **Action on Short Press** [*Left = 0; Right = 1 / Left = 1; Right = 0*]: sets the value that will be sent through "**[Btn] [PX] Two objects - Short press**" after the user short-presses one of the two buttons.

  Note: *this parameter will remain hidden unless having selected "Every button pair is configured separately" in* ***Action of the pair buttons****.*

- **Action on Long Press** [*Left = 0; Right = 1 / Left = 1; Right = 0*]: sets the value that will be sent through "**[Btn] [PX] Two objects - Long press**" after the user long-presses one of the two buttons.

  Note: t*his parameter will remain hidden unless having selected "Every button pair is configured separately" in* ***Action of the pair buttons****.*

Dimmer

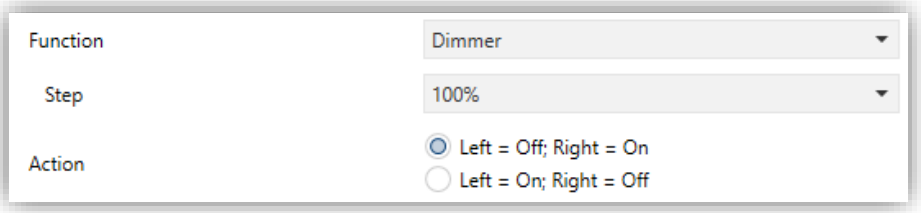

Figure 22. Pair Buttons - Dimmer.

The switch orders will be sent through the "**[Btn] [PX] Light - On/Off**" one-bit object, while the increase/decrease orders will be through the "**[Btn] [PX] Light - Dimming**" four-bit object.

On the other hand, the "**[Btn] [PX] Light - Dimming (Status)**" one-byte object may be linked to the light level status object of the dimmer (in fact, this object is only intended to receive values from the bus, not to send them).

The parameters for this function are:

- **Step** [*100% / 50% / 25% / 12,5% / 6,25% / 3,1% / 1,5%*]: defines the dimming step to be sent to the light dimmer with every increase / decrease order.

  Note: *since dimmers typically do not apply the new light level immediately (i.e., the step regulation is performed progressively) and since capacitive touch switches send an order to interrupt the step dimming once the user releases the button, it is advisable to configure a step of 100%. This way, the user can perform any dimming step by simply leaving the button pressed and then releasing it without needing to make successive button presses.*

- **Action** [*Left = Off; Right = On / Left = On; Right = Off*]: assigns each of the two buttons the order to be sent.

  Note: *this parameter will remain hidden unless having selected "Every button pair is configured separately" in* ***Action of the pair buttons****.*

Shutter

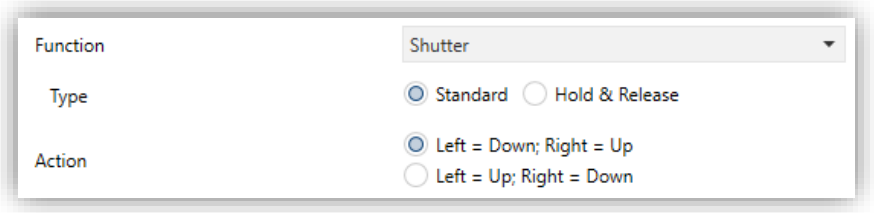

Figure 23. Pair Buttons - Shutter.

The move orders will be sent through "**[Btn] [PX] Shutter - Move**", while the stop orders will be sent through "**[Btn] [X] Shutter Stop/Step**" (for Standard type) or "**[Btn] [PX] Shutter - Stop**" (for Hold & Release type). The parameters for this function are:

- **Type** [*Standard / Hold & Release*]: sets the desired behaviour of the buttons.
- **Action** [*Left = Down; Right = Up / Left = Up; Right = Down*]: assigns each of the two buttons the order to be sent.

  Note: *this parameter will remain hidden unless having selected "Every button pair is configured separately" in* ***Action of the pair buttons****.*

ANNEX I. LED ILLUMINATION MODES

The LED backlight of every button, by default (in most functions), will turn on for a brief instant whenever the button is touched. This behaviour is referred to as the "**Regular Illumination**".

However, in most cases it is possible to assign different behaviours to the LEDs. Which options are available will depend on the function parameterised for the button, but will always include some of the following:

- **Regular Illumination**: the LED will light for an instant once the button is touched.
- **State-Dependent Illumination**: the LED will or will not light, depending on the value of the communication object that corresponds to the function implemented by the button. The exact correspondence between the different values of the object and the different states of the LED may be slightly different from one type of control to another and is detailed for each function.
- **State-Dependent Illumination (both LEDs**): only applies to buttons configured as pair controls. The two LEDs of the control will light or not, depending on the value of the related object and on the particular control type parameterised for that pair of buttons. The only difference compared to the previous case is that, under "both LEDs", the two LEDs will always turn off or on simultaneously, as if it were a unique indicator consisting of two LEDs.
- **Dedicated Object**: the LED will light or not depending on the value ("0" or "1", configurable) of a binary, independent object. In the case of the pair controls, the value "0" will make one of the LEDs light (leaving the other one off), while the value "1" will make them switch their states.

Table 4 illustrates which of the above are configurable for each function.

		Disabled	Regular	State-dep.	State-dep. (both LEDs)	Dedicated object
PAIR	Switch		✓	✓	✓	✓
	Two Objects		✓	✓	✓	✓
	Dimmer		✓	✓	✓	✓
	Shutter		✓			✓
INDIVIDUAL	Switch		✓	✓		✓
	Hold & Release		✓			✓
	Two Objects		✓	✓		✓
	Scene		✓			✓
	Constants		✓	✓		✓
	Dimmer		✓	✓		✓
	Shutter		✓			✓
	LED Indicator					✓
	Room State		✓	✓		✓
DISABLED		✓				

Table 4 Functions vs. LED Illumination Options.

Note:

Regarding the LEDs, it is interesting to distinguish the following cases:

- *Disabled button: the LED will remain off, and the button will have no function.*
- *Button configured as "Individual" with "LED Indicator" function: the button will still have no function. The LED may be turned on/off through a binary object.*
- *Button configured as any other control type: the behaviour of the LED will be configurable according to the following table (being also possible to leave it turned off).*

Although the behaviour of the LEDs can be configured independently for each control, it is also possible to define a **general behaviour for all of them** thus not being then necessary to configure the same option multiple times.

In case of opting for a general configuration, the options are:

- **Regular.**
- **State-Dependent (where available)**. Functions where “state-dependent” is not available will use the regular illumination.
- **State-Dependent (where available) (both LEDs)**. Functions where “state-dependent” is not available will use the regular illumination.
- **Dedicated Object**. One binary communication object per control will be included in the project topology so that the LED of every control turns on/off depending on its own object.

ETS PARAMETERISATION

For details on the parameterisation of the LED illumination modes please refer to the pages that cover the specific function being assigned to the button (see section 2.2).

In case of desiring a **similar behaviour for all of the LEDs**, please find the parameter **LED Illumination Control (All buttons)** in the options of the “General” configuration.

Join and send us your inquiries about Zennio devices:

https://support.zennio.com

Zennio Avance y Tecnología S.L.
C/ Río Jarama, 132. Nave P-8.11
45007 Toledo (Spain).

Tel. +34 925 232 002.

www.zennio.com
info@zennio.com