



# *Touch-MyDesign*

## **KNX Capacitive Touch Switch with 4/6/8 buttons**

**ZN1VI-TPTMD4**

**ZN1VI-TPTMD6**

**ZN1VI-TPTMD8**

Application Program Version: [1.0]

User Manual Version: [1.0]\_a

[www.zennio.com](http://www.zennio.com)

## Contents

1	Introduction .....	3
1.1	Touch-MyDesign.....	3
1.2	Installation.....	4
2	Configuration .....	7
2.1	General Configuration.....	7
2.2	Touch Panel.....	7
3	ETS Parameterization.....	10
3.1	Default Parameterization .....	10
3.2	General .....	11
3.3	Main Buttons.....	18
3.3.1	Pair.....	20
3.3.2	Individual .....	23
3.4	Additional Buttons.....	27
3.4.1	Disabled .....	28
3.4.2	Setpoint Temperature .....	29
3.4.3	1-byte Control (unsigned int) .....	29
3.4.4	1-byte Control (scaling) .....	30
3.4.5	Individual Buttons.....	30
3.4.6	Individual Indicators .....	31
3.5	Inputs.....	32
3.5.1	Push Button .....	33
3.5.2	Switch/Sensor .....	37
3.5.3	Temperature Probe .....	38
3.5.4	Movement Detector .....	40
3.6	Thermostat .....	41
	ANNEX I: Communication objects.....	42

# 1 INTRODUCTION

---

## 1.1 TOUCH-MYDESIGN

---

Touch-MyDesign, the KNX capacitive touch switch from Zennio, is a multifunction and fully customizable solution for room control, including hotel rooms, offices or any other environment where user control is required for climate systems, lighting, blinds, scenes, etc. The versatility of all these functions is enhanced by the built-in analogue/digital inputs, temperature sensor and thermostat function, and by an elegant and fully customizable design of the front glass: consumers can choose the button icons, the texts, the colours and even personalize the background with their pictures, logos, etc.

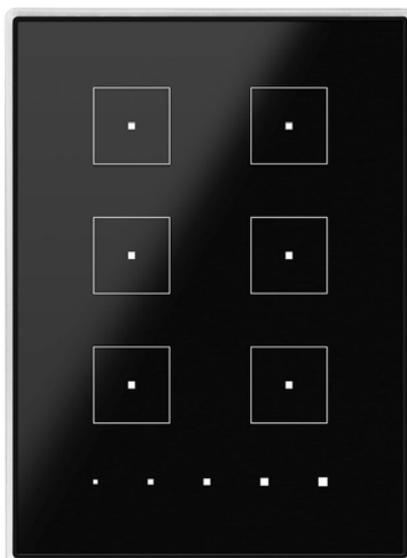


Figure 1 Touch-MyDesign (6-button model)

The most outstanding features of Touch-MyDesign are shown next.

- **Fully customizable** design of the front glass.
- **4 / 6 / 8 main touch buttons** (configurable individually or in pairs).
- **5 additional touch buttons** (configurable individually or jointly).
- **Horizontal or vertical** orientation.

- **Light indicator (LED)** for every button or control, with the possibility of making the light status depend on the control status.
- **Buzzer** for an audible acknowledgement of user actions, with the possibility of disabling it either by parameter or by object.
- Possibility of **locking / unlocking the touch panel** through binary orders or scenes, and of setting a timed/automatic locking of the device.
- **Welcome Back object** (binary or scene), which will be sent to the bus on the first press of the touch panel after a long standby period.
- **Two analogue/digital inputs** (for movement sensors, temperature probes, additional switches, etc.).
- **Thermostat** function.
- Built-in **temperature sensor**.

## 1.2 INSTALLATION

---

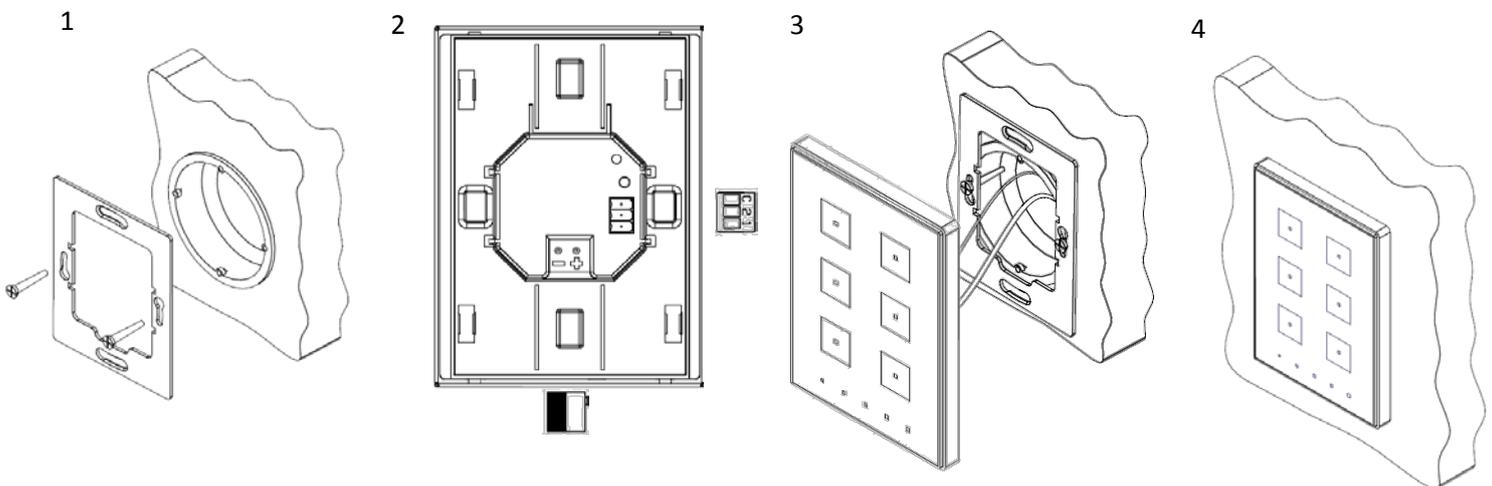


Figure 2 Touch-MyDesign. Installation process.

To install the device, it is first necessary to mount the metallic plate into a square/round standard appliance box through the corresponding screws. Next, Touch-MyDesign is connected to the KNX bus through the corresponding terminal on the rear side of the device, and then the input terminal is as well connected to the rear of the device.

Once the input terminal and the KNX terminal are connected, the device can be easily mounted on the metallic plate by the action of the built-in magnets. After that, it is necessary to slide Touch-MyDesign downwards to fix it through the security anchorage system.

Finally, it is advisable to check that the device is properly installed, and that only the profile of the device becomes visible from above, from below and from both sides (the metallic plate should be completely hidden).

This device does not need any external power supply since it is powered through the KNX bus.

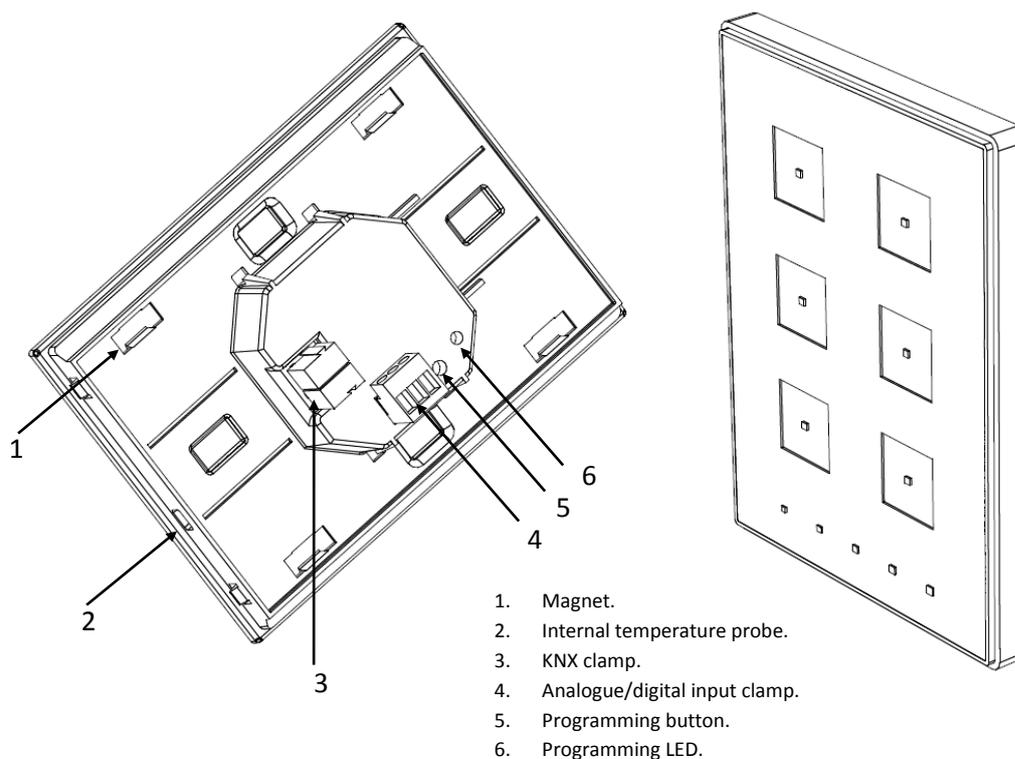


Figure 3 Touch-MyDesign. Element diagram.

The programming button (5) shown in Figure 3 may be pressed with the help of a thin screw to set Touch-MyDesign in **programming mode**. After a short press, the programming LED will light in red. Note that if this button is held while plugging the device into the KNX bus, the device will enter the **secure mode**. The LED will then blink in red.

**Note:** whenever the device recovers the bus power, an immediate self-calibration process of the touch panel takes place. Please ensure to avoid making pressure over the front glass while powering the device. If undesired effects arise during use, please

*disconnect the device from the bus and connect it again, making sure that the front glass is not touched during this process.*

For detailed information about the technical features of Touch-MyDesign, as well as information on security and on the installation process, please refer to the **Datasheet** included within the original packaging of the device and also available at <http://www.zennio.com>.

## 2 CONFIGURATION

---

### 2.1 GENERAL CONFIGURATION

---

Zennio Touch-MyDesign lets the user control and monitor a set of functionalities within a domotic environment, in an easy and intuitive manner. The inexistence of a screen, menus or complex user interaction beyond button presses confers the device a notable ease of use.

To make the device perform the desired functions, several options need to be parameterized, both in relation to the **general behaving** (horizontal/vertical orientation, locking procedure of the touch panel, buzzing for action confirmation, welcome back object...) and **button-specific** (function to be performed, behaviour of its corresponding LED, etc.).

On the other hand, Touch-MyDesign features 2 opto-coupled inputs, each of which may be independently configured as a **switch/sensor**, a **push-button**, a **movement detector**, or a **temperature probe**. And depending on the selection, a series of external elements may be connected to the input terminal of Touch-MyDesign. In the particular case of an external temperature probe (such as model **ZN1AC-NTC68** from Zennio), it will be possible to use it independently of the temperature sensor built-in in the device, which implements its own communication objects and may or may not be enabled by parameter.

### 2.2 TOUCH PANEL

---

Touch-MyDesign features **4, 6 or 8 'main' capacitive buttons** (depending on the model) at the user's disposal for the execution of actions. **5 more 'additional' buttons** can be found at the bottom of the front touch panel (or at the right side of the front touch panel, in case of mounting the device horizontally). All of them will perform specific and permanent functionalities at any time, since functions are not grouped into alternating menus, pages...

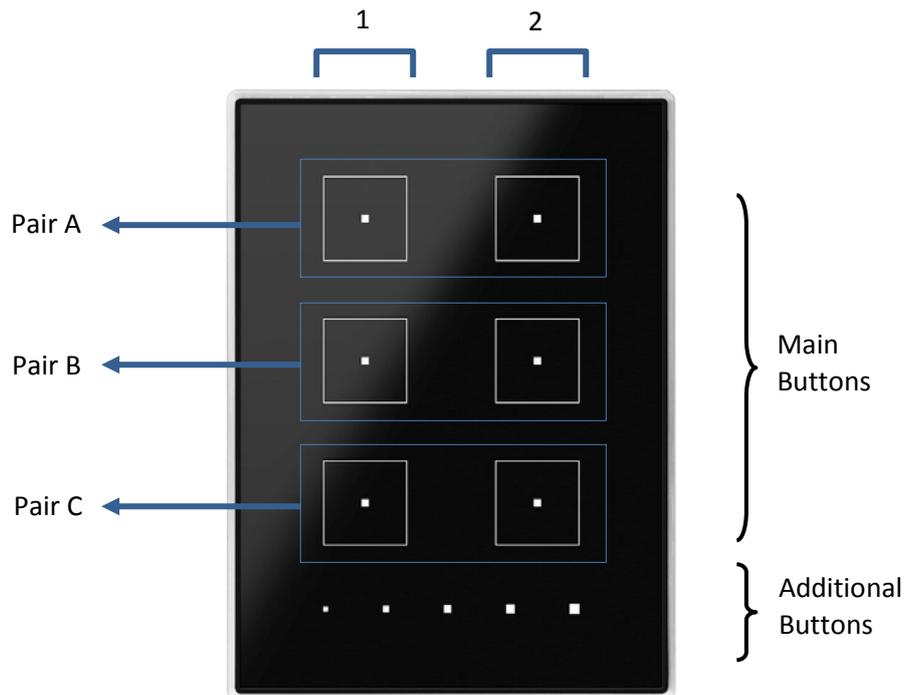


Figure 4 Touch panel (6-button model)

As Figure 4 shows, two areas can be distinguished in the touch panel:

- **The main buttons**, grouped in pairs, all over most of the touch panel surface.
- **The additional buttons**, as an aligned panel, separate from the main buttons.

All the main buttons are identical, and also every additional button is identical to the others, which makes it possible to configure all of them for user-defined applications.

Each of the mentioned buttons incorporates a central LED which, by default, will turn on for a brief instant whenever the button is touched, although alternative LED behaviours may be configured for every button, depending on the parameterized function:

- **Regular lighting**: the LED will light for only an instant after the button is touched. LEDs behave this way by default.
- **State-dependent lighting**: 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 relation 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, so it will be explained in later sections.

- **State-dependent lighting (both LEDs):** only applies to main buttons that are configured as pair-button controls. The two LEDs of the pair-button control will light or not depending on the value of the related object and on the particular control type parameterized for that pair of buttons. The only difference compared to the previous case is that, under the “both LEDs” case, the two LEDs will always turn off or on simultaneously, as if it were a unique state indicator consisting of two LEDs.

Apart from the behaviour of the LEDs, **beeping** can be activated or deactivated as an acoustic feedback for the user when an action is performed after a button touch. Enabling and disabling the buzzer can be done by parameter or by object, being also possible to define by parameter whether this function should be initially enabled or not. Finally, a specific object has also been included for externally triggering a brief beep at any time, provided that the beeping function has not been disabled.

## 3 ETS PARAMETERIZATION

To begin with the parameterization process of the device, it is necessary, once the ETS program has been opened, to import the database of the product (**Touch-MyDesign** application program).

Next, the device should be added to the project where desired. And then, one right-click on the device will permit selecting "Edit parameters", in order to start the configuration.

In the following sections a detailed explanation can be found about each of the different functionalities of Touch-MyDesign in ETS.

### 3.1 DEFAULT PARAMETERIZATION

This section shows the default configuration from which the device parameterization starts.

Figure 5 shows the communication objects displayed by default: "[General] Scene: receive" (intended for the reception of scene values from the bus), "[General] Scene: send" (destined to send scene values to the bus), "[General] Buzzer enabling" (for enabling or disabling the beeping functionality, which may also be set by parameter, although disabling it by parameter will have a permanent effect and will hide this object) and "[General] Buzzer" (which makes it possible to externally triggering a brief beep at any time –beyond the usual behaviour of these beeps as action confirmations– whenever the state of the "[General] Buzzer enabling" object permits it).

Number	Name	Object Function	Description	Group Addresses	Length
52	[General] Scene: receive	0-63 (Run Scene 1-64)			1 Byte
53	[General] Scene: send	0-63 (Run or Save Scene 1-64)			1 Byte
60	[General] Buzzer	1=Beep; 0=Nothing			1 bit
61	[General] Buzzer Enabling	1=Enabled; 0=Disabled			1 bit

Figure 5 Default topology

When entering the parameter edition of Touch-MyDesign for the first time, a window similar to Figure 6 will be shown, where three main sections are available: **General**, **Main buttons** and **Additional buttons**, which are described next.

### 3.2 GENERAL

The windows under the “General” tab permits configuring the basics of the device. The only of them that is active by default is “Configuration”.

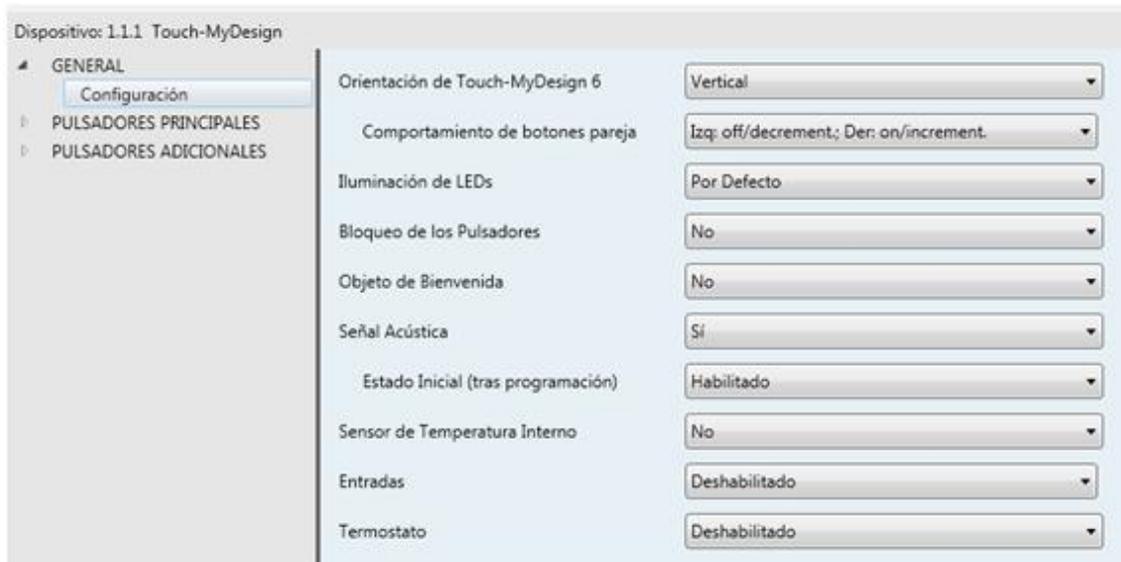


Figure 6 Configuration (General)

- **Touch-MyDesign orientation.** Defines the orientation (horizontal / vertical) of the device, with the aim of implementing a logical behaviour in the buttons.

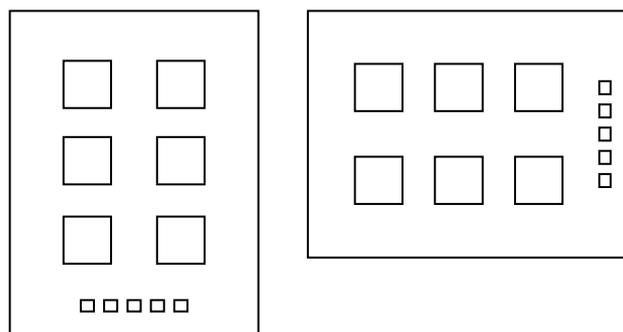


Figure 7 Orientation of the device

Next to “Touch-MyDesign Orientation”, a second parameter is shown, “**Pair button behaviour**”. The selectable values for this parameter will depend on the selection made for the former parameter. Table 1 shows the possible behaviours of the pair buttons. The option selected here will affect every button configured as a pair control. Note that if the device is mounted with a layout inverted to what is

shown by Figure 7 (i.e., with the additional buttons on the left or on the top), the meaning of the words “left” and “right” should be assumed inverted as well.

**Note:** *this parameterization does not alter the names of the touch buttons or of their objects. It does not affect the behaviour of the non-pair buttons (such as the additional buttons) either.*

Orientation	Pair Button behaviour
Vertical	Left: off / decrease Right: on / increase
	Left: on / increase Right: off / decrease
Horizontal	Down: off / decrease Up: on / increase
	Down: on / increase Up: off / decrease

Table 1 Orientation vs. pair button behaviour

● **LED lighting.** In addition to the specific behaviour of every LED regarding its own button (see section 2.2), it is possible to define certain general attributes that will affect all the LEDs. For that purpose, this parameter provides two options:

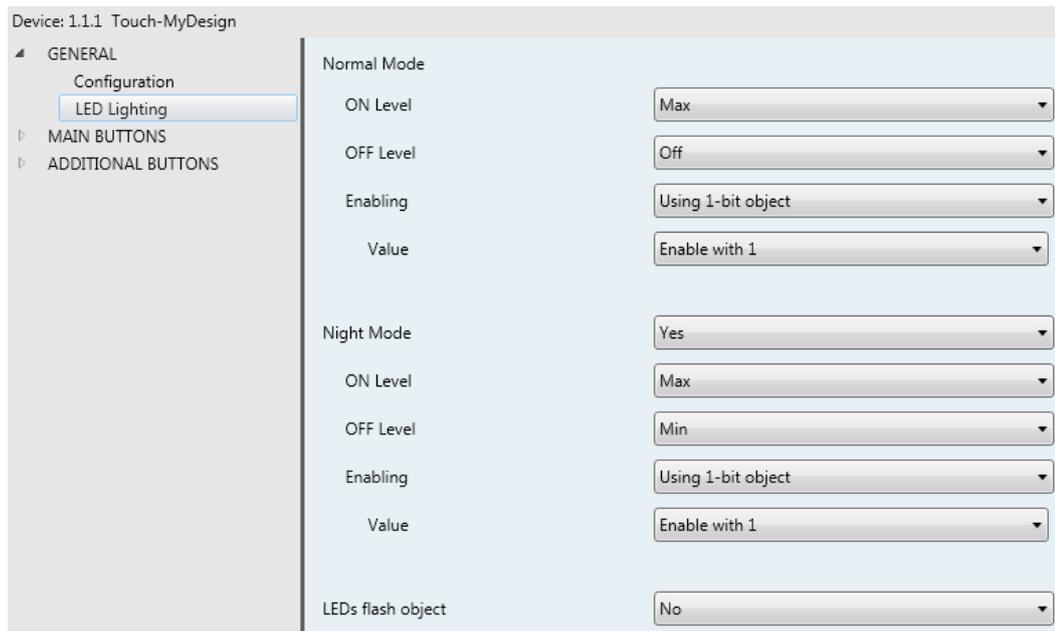
**Default:** the LEDs will assume the default light levels for the “on” and “off” states (maximum level and minimum level, respectively).

**Custom:** permits the definition of custom light levels for the “on” and “off” states of the LEDs, as well as enabling complementary functionalities, like the night mode or the LED flashing object. For that purpose, after selecting this option a new screen will come up (**LED lighting**; see Figure 8), where the following parameters can be found:

- **Normal mode:** this section is provided for defining the desired lighting level (“max” or “min”) for the “on” state of the LEDs under normal condition, and, analogously, the desired level (“off”, “min” or “max”) for the “off” state of the LEDs, under normal condition. The default values are respectively “max” and “off”.

In addition to this normal mode it is also possible to implement a night mode with its own light parameterization (equal or different to that of

the normal mode), so that the device can switch from one mode to another upon a certain event (if the “night mode” functionality has been enabled, the “Activation” parameter will be displayed, among the aforementioned parameters of the “normal mode”, as described below). Note, however, that an equal parameterization for both modes does not mean an equal effective light level; e.g., “on=max” will imply a higher luminosity for normal mode than for night mode. Night mode implements by itself an additional attenuation of the light levels.



**Figure 8** Lighting with the night mode and the flash object enabled

➤ **Night mode:** permits enabling (“Yes”) or disabling (“No”) the functionality of the “night mode”, i.e., a light level configuration complementary to the one of the “normal mode”. The “night mode” functionality is disabled by default. When enabled, two binary objects are shown: “[General] LEDs: night mode” and “[General] LEDs: normal mode”, as well as the following parameters:

- **“ON level”:** defines, for the night mode, the luminosity level (“max” or “min”) that will be assumed for the “on” state of the LEDs.
- **“OFF level”:** defines, for the night mode, the luminosity level (“off”, “min” or “max”) that will be assumed for the “off” status of the LEDs.

- **“Activation”**: this parameter (as well as the analogous parameter for the normal mode that comes up after enabling the night mode function) permits selecting the desired procedure for switching between the normal and the night modes and vice versa, which can consist in the reception of a certain binary value (“0” or “1”, configurable) through the object “[General] LEDs: night mode” (or “[General] LEDs: normal mode”, for normal mode), or in the reception of a certain scene (1 to 64, configurable) through the “[General] Scene: receive”. Note that, being these modes mutually exclusive, the device will leave one mode as soon as it receives the order to enter the other mode.

Note, as already stated, that night mode implements by itself a general attenuation of the light levels. For example, the “max” light level in the normal mode will be brighter than the “max” light level in the night mode. Even the “min” light level in the normal mode will be.

**Note:** *due to hardware reconfiguration, switching from one mode to the other causes that for 2-3 seconds the touch buttons do not react to presses. The application program and the bus order reception are, however, not affected.*

- **“LEDs flash object”**: permits, by selecting “Yes”, making use of a binary object (“[General] Flash LEDs”) so that when the value “1” is received, every LED in the device will turn into the “on” state and maintain it during a certain time (configurable through the “Flash Time” parameter, which accepts input values between 1 and 20 seconds; note that the flash time counter will be reset if the value “1” is received again during the flash time), after which it will recover the corresponding state. This function is not enabled by default.

**Note:** *during the flash time, the LEDs will remain on, ignoring their usual behaviour. However, the device will still react to presses and bus orders, so when the flash time ends the LEDs will acquire their corresponding states.*

**Note:** *the flash function is interrupted on bus failures.*

• **Touch locking.** This function permits locking (i.e., making them useless) or unlocking the touch buttons of the touch panel, with the possibility of setting the following fields (from the “Touch locking” screen, which will become visible after enabling this function; see Figure 9):

**Locking method:** defines the desired procedure for entering into the “locked” state. This can consist in a 1-bit object (“[General] Touch Lock”, which will lock the touch buttons when it receives the values “1” or “0”, according to the parameterization), the reception of a scene (values 1 to 64, configurable) through “[General] Scenes: receive”; or an Auto (timed) triggering which will require setting (through the “Time to lock” parameter, in seconds) how much time the device must stay idle since the last button touch before entering into the “locked” state.

**Unlocking method:** defines the procedure for leaving the “locked” state of the touch panel. This can consist in a 1-bit object (“[General] Touch Unlock”, which will proceed with the unlock on the arrival of the values “1” or “0”, according to parameterization) or in the reception of a scene (values 1 to 64, configurable) through “[General] Scenes: receive”.

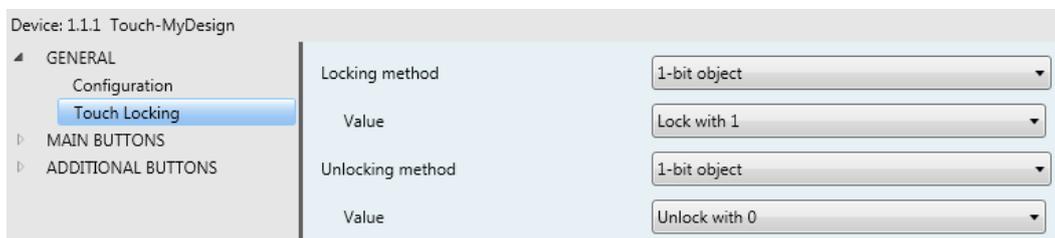


Figure 9 Touch locking (General)

• **Welcome Back object.** This function permits sending a certain value (binary or scene) to the KNX bus when the user performs a touch on any of the touch buttons after a standby period. This makes possible, for example, lighting up a room after several hours of darkness with only a random touch over the touch panel. If enabled, this function will make the “[General] Welcome Back object” object visible, as well as a new window (see Figure 10) from where it is possible to parameterize its behaviour.

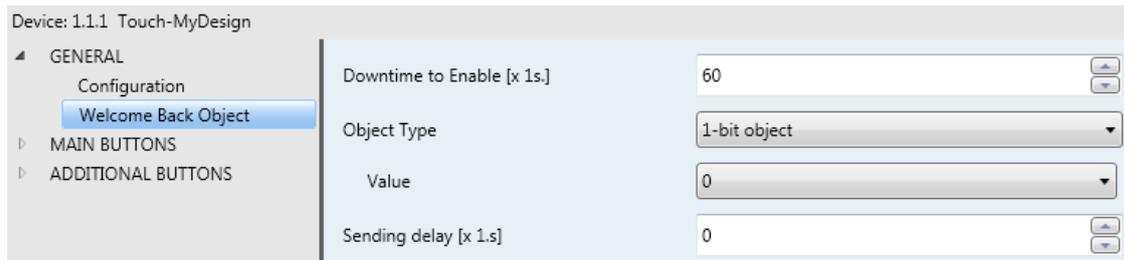


Figure 10 Welcome Back object (General)

**Downtime to enable:** defines the minimum time that must pass since the last touch before the device assumes that the next touch will be associated to the sending of the welcome back object. The allowed values are [5 – 65535] seconds.

**Object type:** this parameter selects the type of sending that is desired to be performed. By default, it will be a 1-bit value (“0” or “1”, configurable), but it is also possible to select “Scene” (1 – 64, configurable). In the case of selecting the latter, the “[General] Welcome back object” object will become hidden, since the sending will be performed through the “[General] Scene: send” object.

**Sending delay:** time (between 0 and 255 seconds) that Touch-MyDesign will wait since the touch takes place and the welcome back object is effectively sent.

**Note:** *if combining the welcome back function with the automatic (timed) touch lock function, it is important to take into account that the delay of the sending of the welcome back object should not exceed the timed self-lock. For instance, if an automatic lock after 60 seconds has been parameterized, the welcome back object will actually not be sent if the sending delay exceeds those 60 seconds (the device will lock before the sending is performed). This is particularly important if linking the welcome back object to the unlocking object.*

● **Buzzer:** permits enabling or disabling the audible beeps that the device can generate. Disabling them by parameter entirely prevents their use, while enabling them, apart from activating an audible acknowledgment for the actions triggered by button touches (see section 2.2), will also bring up the objects “[General] Buzzer” (which permits externally triggering the generation of a short beep at any time by sending the value “1”) and “[General] Buzzer enabling” (which permits

enabling and disabling, by object, any audible indication generated by the buzzer, including those due to the sending of a “1” through “[General] Buzzer”). By default, beeping is enabled both by parameter and by object. However, even being enabled by parameter, it is also possible to define (through the “Initial status (after programming)” parameter) the desired initial state of the “[General] Buzzer enabling” object. Note, however, that if the buzzer is disabled by parameter, the two objects related to this function will disappear, which will make it impossible to enable it afterwards.

● **Internal temperature sensor:** enabling (“Yes”) this parameter will bring up a new object (“[Internal sensor] Current temperature”), as well as a new parameterizable window (see Figure 11), from where it is possible to define the following:

Figure 11 Internal temperature sensor (General)

**Temperature sensor calibration:** this option permits implementing a correction (by adding or subtracting some tenths of a degree) to the measured temperature in case of a deviation between it and the real temperature in the room. The allowed values are [-50, 50], i.e., from -5 to 5 °C.

**Temperature sending period:** permits performing a cyclical sending of the temperature value to the KNX bus through the corresponding object. This parameter is intended to define the desired cycle time, between 1 and 100 tens of a second. The value “0” will turn off the cyclical sending.

✓ **Example:** *to have the temperature of the internal sensor sent every 30 seconds, this parameter should be set to “3”.*

**Send with a temperature change:** permits performing an automatic sending of the temperature value whenever it suffers a change (an increase or a decrease) greater than a certain amount of degrees, which should be defined through this parameter (in tenths of a degree). The allowed values are “0” (disabled) to “200” (20°C). Note that this function is independent from the cyclical sending already described.

✓ **Example:** *to have the temperature sent to the bus whenever a change of 5°C or more is detected between two consecutive measurements, the value “50” is used.*

**Temperature protection:** the internal temperature sensor implements a protection function against overheating, overcooling or overheating and overcooling which may be enabled through this parameter. Depending on the selected type of protection, one or two binary objects will become visible: “[Internal sensor] Overheating” and “[Internal sensor] Overcooling”, which will send the value “1” as soon as the temperature reaches the limit corresponding to each case, and the “0” as soon as such situation ends. It is therefore necessary to define the limit temperature (in Celsius degrees) for overheating, for overcooling, or for both. Moreover, it is possible to define a hysteresis (from 1 to 80 tenths of a degree) to avoid a consecutive re-sending when the temperature fluctuates around the defined limit.

● **Inputs:** permits enabling and configuring each of the two inputs available in the device (see section 2.1), individually. For that purpose, after selecting the value “Enabled”, ETS will bring up a new tab with the name “Inputs”, which will be explained in detail in section 3.5.

● **Thermostat:** permits enabling and configuring the thermostat function. For that purpose, after selecting the value “Enabled”, ETS will bring up a new tab with the name “Thermostat”, which will be described in detail in section 3.6.

### 3.3 MAIN BUTTONS

---

This tab, which permits defining the specific functions that will be performed by the main buttons of the device (see section 2.2), is itself divided into a series of windows, among which the only one visible by default is Configuration.



Figure 12 Configuration (Main buttons)

From the Configuration window it is possible to assign to every pair of buttons (named A, B and C, in the case of Touch-MyDesign 6) a behaviour as such (option “Pair”, which will bring up a new parameterizable window with the name “Pair X”, where X will be “A”, “B” or “C”), or a mutually independent behaviour of its two buttons (option “Individual”, which will bring up two new parameterizable windows with the names “Button X1” and “Button X2”, where X will be “A”, “B” or “C”). It is also possible to entirely disable each pair of buttons (and its LEDs) through the option “Disabled”.

In short, the following functions are available for the main buttons, depending on the behaviour (“Disabled”, “Pair” or “Individual”) parameterized for each pair of buttons:

- **Disabled** (non functional buttons; inactive LEDs).
- **Pairs:**
  - Switch.
  - Dimmer.
  - Shutter.
- **Individual:**
  - Disabled.
  - 1 bit.
  - 1 bit (hold & release).
  - Scene.
  - 1-byte constant (unsigned int).
  - 1-byte constant (scaling).
  - 2-byte constant (unsigned int).
  - 2-byte constant (float).
  - Dimmer.
  - Shutter.

The options under “Pair” and “Individual” are explained in detail in sections 3.3.1 and 3.3.2.

**Note:** *disabling a pair of buttons from the “Configuration” window implies that none of the buttons and the LEDs in the pair will be functional. Alternatively, it is possible to select the option “Individual” for the pair of buttons and, afterwards, actually setting or not a functionality to each of them (and/or to its respective LED) from the “Button” window (see section 3.3.2).*

### 3.3.1 PAIR

In the case of selecting the option “Pair” in any of the drop-down lists from the “Configuration” page of the “Main buttons” tab, a new page (“Pair A”, “Pair B”, “Pair C”, as corresponding) from where it will be possible to set the desired functionality of the buttons and LEDs that constitute the corresponding pair.

**Note:** figures in this section may refer to the case of “Pair A”. Any other cases should be analogous.



Figure 13 Pair A (Main buttons)

The parameters that are shown here by default are Function (with the options “Switch”, “Dimmer” and “Shutter”) and LED Lighting, being the latter conditioned by the option that has been selected for the former, as described next:

- **Switch:** selecting this option brings up the (“[X] Binary control”) object, through which the values “0” or “1” will be sent to the KNX bus depending on user touches over the buttons of the pair. The particular correspondence between each button in the pair and the value sent is subject to the general parameters “Touch-MyDesign orientation” and “Pair button behaviour” (see section 3.2).

When the “Switch” option has been selected, the parameter **LED lighting** permits, for its part, three alternatives:

**Regular:** the LED of each button in the pair will behave in the usual way: when a touch on its button is detected, it will light (according to the custom “ON” level, in parameterized) for an instant and it will then turn off again (according to the custom “OFF” level, if parameterized).

**State-dependent:** the light state of the LEDs in the pair will depend on the current value of the “[X] Binary control” object, so there will always be one of the LEDs in the “ON” status (but never both at the same time). This way, the LEDs will behave as a state indicator (one of the LEDs will light

when the object is “On”, and the other one will do when the object is “Off”).

**State-dependent (both LEDs):** the light status of the LEDs in the pair will be, as above, determined by the value of the “[X] Binary control” object, therefore behaving as a state indicator. However, in this case both LEDs will always share the same light status, i.e., switching on and off together.

- **Dimmer:** selecting this option, which permits implementing light dimming through the pair of buttons, will bring up two new communication objects: “[X] Light On/Off” (binary object for switching on/off the light source, by sending the values “1” or “0”, respectively) and “[X] Light dimming” (a 4-bit object that permits performing step-dimming of the light source, according to Table 2).

Dimming step	Number of presses required for the entire dimming (0 – 100%)
(1) 100%	1
(2) 50%	2
(3) 25%	4
(4) 12.5%	8
(5) 6.25%	16
(6) 3.1%	32
(7) 1.5%	64

Table 2 Step-dimming

Pair buttons configured for light dimming behave as follows:

**Off/Decrease button:** a short touch will send an order to turn the light off (“[X] Light On/Off”=0). A long touch will send an order to decrease the light level the amount parameterized as the dimming step, unless the button is released again before this decrease ends, which will send a “stop” order (0x00, in this case).

**On/Increase button:** a short touch will send an order to turn the light on (“[X] Light On/Off”=1). A long touch will send an order to increase the light level the amount parameterized as the dimming step, unless the button is released again before this increase ends, which will send a “stop” order (0x08, in this case).

When the “Dimmer” option has been selected, the parameter **LED lighting** permits, for its part, three alternatives:

**Regular:** the LED of each button in the pair will behave in the usual way: when a touch on its button is detected, it will light (according to the custom “ON” level, in parameterized) for an instant and it will then turn off again (according to the custom “OFF” level, if parameterized).

**State-dependent:** the light state of the LEDs in the pair will depend on the current value of the “[X] Light On/Off” object, so there will always be one of the LEDs in the “ON” status (but never both at the same time). This way, the LEDs will behave as a state indicator (one of the LEDs will light when the object is “On”, and the other one will do when the object is “Off”).

**State-dependent (both LEDs):** the light status of the LEDs in the pair will be, as above, determined by the value of the “[X] Light On/Off” object, therefore behaving as a state indicator. However, in this case both LEDs will always share the same light status, i.e., switching on and off together.

● **Shutter:** this option permits controlling shutters by means of two 1-bit objects, “[X] Move shutter” (“0”=Up and “1”=Down) and “[X] Stop shutter / Step” (both “0” and “1” will interrupt the movement). The behaving of these buttons is as follows:

**“Decrease” button:** a long touch will send an order to move the shutter down, whereas a short touch will send the value “1” through the “[X] Stop shutter / step” object, which will make the shutter stop (if moving) or perform a short step downwards.

**“Increase” button:** a long touch will send an order to move the shutter up, whereas a short touch will send the value “0” through the “[X] Stop shutter / step” object, which will make the shutter stop (if moving) or perform a short step upwards.

When the “Shutter” option has been selected, the parameter **LED lighting** will not be shown anymore, since the LEDs acquire the already-defined “regular” behaviour.

### 3.3.2 INDIVIDUAL

In the case of selecting the option “Individual” in any of the drop-down lists from the “Configuration” page of the “Main buttons” tab, two new pages (“Button X1” and “Button X2”, where X is the letter of the pair for which the “Individual” option has been selected) from where it will be possible to set the desired functionality of the buttons and LEDs that constitute the corresponding pair.

**Note:** figures in this section may refer to the case of “Button A1”. Any other cases should be analogous.



Figure 14 Button A1 (Main buttons)

The parameters that are shown here by default are **Function** and **LED Lighting**, being the latter conditioned by the option that has been selected for the former, which permits the following options:

- **Disabled:** this option (disabled by default) entirely disables the functionality of button Xi, however there will still be the possibility of selecting (through the “LED indicator” parameter) whether the associated LED behaves regularly (“Active”) and therefore turns on for an instant whenever a press happens on the button (and afterwards turns off), or remains totally inactive.
- **1-bit:** permits sending a certain binary value, B (“0”, “1” or an alternating value), through the “[Xi] Binary control: B” object, visible after assigning this functionality to button Xi.

The parameter **LED lighting** permits, for its part, two alternatives for the behaviour of the LED corresponding to the button:

**Regular:** the LED will behave in the usual way: when a touch on its button is detected, it will light (according to the custom “ON” level, in parameterized) for an instant and it will then turn off again (according to the custom “OFF” level, if parameterized).

**State-dependent:** the light state of the LED will depend on the current value of the “[Xi] Binary control: B” object, so it will remain in the “ON” state whenever the value of the object is “1” and in the “OFF” state when the value is “0”. Note that, in this case, the described behaviour does not depend on the parameterized value B (i.e., on the value that is sent to the bus when the button is touched).

**Example:** *assume that, from the General parameter window, a custom lighting has been set, consisting in an ON level equal to “maximum” and an OFF level equal to “minimum”. Also assume that a certain button has been programmed as a switched 1-bit control, and, for its respective LED, the “state-dependent” mode has been selected. In such case: caso:*

- *Whenever the object that corresponds to the button has the value “1”, the LED will light with the maximum power level.*
- *Whenever the object that corresponds to the button has the value “0”, the LED will light with the minimum power level.*
- *Pressing the button will successively commute (0-1-0-1-0...) the value of the object.*

● **1-bit (hold & release):** permits sending a certain binary value, B (“0”, “1”, parameterizable), through the “[Xi] Binary control, Hold: B” object as soon as the button is touched, and another binary value, B’ (different from B or not) through the “[Xi] Binary control, release: B” as soon as the press ends.

When the “1-bit (hold & release)” option has been selected, the parameter **LED lighting** will not be shown anymore, since the LED acquires the already-defined “regular” behaviour.

● **Scene:** permits that short touches on the button trigger the sending of a scene (1 – 64, configurable) to the KNX bus through the “[General] Scenes: send” object, and, optionally, long touches (3-second or longer touches, in this case) trigger as well the sending of a scene recording order, so when such long

touches take place, the original scene (that is, the one which corresponded to the configured number, from 1 to 64) is replaced by a new one.

When the “Scene” option has been selected, the parameter **LED lighting** will not be shown anymore, since the LED acquires the already-defined “regular” behaviour.

- **1-byte constant (unsigned int):** permits that short touches on the button trigger the sending of a certain 1-byte unsigned integer value (0-255) to the KNX bus through the “[Xi] 1-byte value (unsigned int)” object.

The parameter **LED lighting** permits, for its part, two alternatives for the behaviour of the LED corresponding to the button:

**Regular:** the LED will behave in the usual way: when a touch on its button is detected, it will light (according to the custom “ON” level, in parameterized) for an instant and it will then turn off again (according to the custom “OFF” level, if parameterized).

**State-dependent:** the light state of the LED will depend on the current value of the “[Xi] 1-byte value (unsigned int)” object, so it will remain in the “ON” state whenever the value of the object equals the parameterized unsigned integer number (that is, the value that is sent to the bus on button touches) and in the “OFF” state when the value of the object does not.

- **1-byte constant (scaling):** permits that short touches on the button trigger the sending of a certain percentage value (0-100) to the KNX bus through the “[Xi] Scaling” value.

The parameter **LED lighting** permits, for its part, defining the behaviour of the LED corresponding to the button. The parameterization is analogous to that already described for the previous case, although the lighting state depends here on the “[Xi] 1-byte value (scaling)” object.

- **2-byte constant (unsigned int):** permits that short touches on the button trigger the sending of a certain 2-byte unsigned integer value (0-65535) to the KNX bus through the “[Xi] 2-byte value (unsigned int)” object.

The parameter **LED lighting** permits, for its part, defining the behaviour of the LED corresponding to the button. The parameterization is analogous to that already described for the previous case, although the lighting state depends here on the “[Xi] 2-byte value (unsigned int)”.

- **2-byte constant (float)**: permits that short touches on the button trigger the sending of a 2-byte floating point value (-25.0 to 90.0) through the “[Xi] 2-byte value (float)” object.

The parameter **LED lighting** permits, for its part, defining the behaviour of the LED corresponding to the button. The parameterization is analogous to that already described for the previous case: “Regular” (the LED will stay “ON” for an instant when the button is touched) or “State-dependent” (the LED will stay “ON” when the value of the “[Xi] 2-byte value (float)” object equals the numeric value parameterized, and “OFF” in any other case).

- **Dimmer**: permits performing light system dimming by using a sole touch button and two communication objects: “[Xi] Light On/Off (Toggle)” (1-bit, linked to short touches) and “[Xi] Light dimming (Toggle)” (4-bit, linked to long touches).

Having only one button to perform these functions implies commuted responses on button touches, as in the following example:

✓ **Example**: *after the first touch on the button, a switch-on order (“1”) will be sent through the “[Xi] Light On/Off” object. A later short touch on the same button will cause the sending of a switch-off (“0”) order through the same object. After that, a further short touch will cause the sending of the value “1” again, and so on with the following touches.*

Also step dimming implements a commuted behaviour – if a long touch caused the sending of an order to increase the light level, the next touch will send an order to decrease it; and analogously for the inverse case. However, after a switch-off through a short touch, the next long touch will always imply an order to increase the light. Similarly, if the last (short) touch caused a switch-on, the next long touch will necessarily cause a decrease in the light level.

The parameter **LED lighting** permits, for its part, defining the behaviour of the LED corresponding to the button. As for the previous cases, the available options

are: “Regular” (the LED will stay “ON” for an instant when the button is touched) or “State-dependent” (the LED will stay “ON” when the value of the “[Xi] Light On/Off (commuted)” object is “1”, and “OFF” if not).

● **Shutter:** permits controlling shutters by means of a sole touch button. For this functionality no further parameterization is required. Once selected, the object “[Xi] Move shutter (toggle dir.)” associated to long touches and the object “[Xi] Stop shutter / step (toggle dir.)” associated to short touches will come up. Both are commuted binary objects and will behave independently from each other, as in the following example:

✓ **Example:** after a short touch on the button, a downwards step order (“1”) will be sent through the “[Xi] Stop shutter / step (toggle dir.)” object. A later short touch on the button will cause the sending of an upwards step order (“0”) through the same object. After that, a further short touch will cause the sending of the value “1” again, and so on for the next touches. Moreover, if a long touch is performed at any time, an order to move the shutter entirely down (“1”) will be sent through the “[Xi] Move shutter (toggle dir.)” object. The next long touch will send an order to move it entirely up (“0”) through the same object). A third long touch will cause the sending of the value “1” again, and so on for further long touches.

In this case, the LED corresponding to the button will always implement the already-defined “Regular” behaviour (the LED will stay “ON” for some instants after the press).

### 3.4 ADDITIONAL BUTTONS

This tab, which permits defining the specific functions that will be performed by the additional buttons of the device (see section 2.2), is itself divided into a series of windows, among which the only one visible by default is Configuration.



Figure 15 Configuration (additional buttons)

From the Configuration window it is possible to select (parameter “Function”) the desired behaviours for the five buttons that constitute the additional button panel, according to the following list:

- **Disabled**
- **Temperature Setpoint**
- **1-byte control (unsigned int)**
- **1-byte control (scaling)**
- **Individual buttons:**
  - Disabled
  - 1-bit
  - 1-bit (hold & release)
  - Scene
  - 1-byte constant (unsigned int)
  - 1-byte constant (scaling)
  - 2-byte constant (unsigned int)
  - 2-byte constant (float)
  - Dimmer.
  - Shutter.
- **Individual indicators** (touching the button will have no effect, although the LEDs will work as binary state indicators).

Regarding the above options, “Individual buttons” permits configuring each of the additional buttons as an independent control (analogously as with any of the main buttons), while all the other options are intended to jointly parameterize a certain functionality that requires the five additional buttons all together.

The parameterization of each of the above alternatives is detailed in Section 3.4.1 and the subsequent sections.

### 3.4.1 DISABLED

---

Disabling the additional button panel from the “Configuration” screen within the “Additional buttons” tab will cause that none of the additional buttons has functionality. However, through the “**LED Lighting**” parameter, it is possible to select whether the corresponding LEDs should always remain in the off state (“Inactive”) or acquire the “regular” behaviour (“Active”) thus acquiring the “on” light state for some instants after a button touch, although this will not have a practical consequence.

### 3.4.2 SETPOINT TEMPERATURE

This option permits implementing a thermostatic control through the five additional buttons all together. When enabled, a new window (“Setpoint temperature”) will be displayed within the “Additional buttons” tab, from where a different setpoint may be parameterized for each button. These values are intended to be sent to the KNX bus (through the “[Z] Temperature setpoint”) depending on the additional touch button being touched.

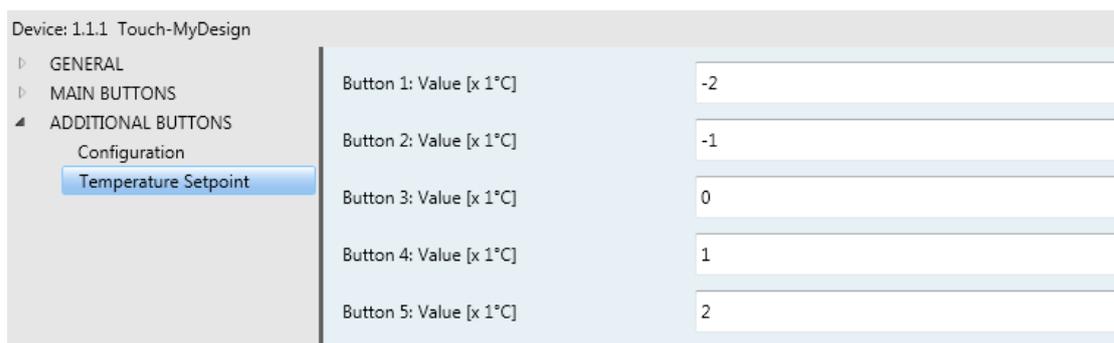


Figure 16 Temperature setpoint (additional buttons)

This type of functionality will always entail a **“state-dependent”** behaviour in the LEDs: as soon as the “[Z] Temperature setpoint” object receives a value that equals one of the parameterized values for the additional buttons, the LED corresponding to the matching button will turn on, while the other LEDs will turn off.

### 3.4.3 1-BYTE CONTROL (UNSIGNED INT)

This option permits parameterizing the five additional buttons all together in order to implement a joint 1-byte control so that, on the event of one of them being touched, the KNX bus will be sent (through the “[Z] 1-byte value (unsigned int)” object) a certain value (0-255), depending on the particular button being touched.

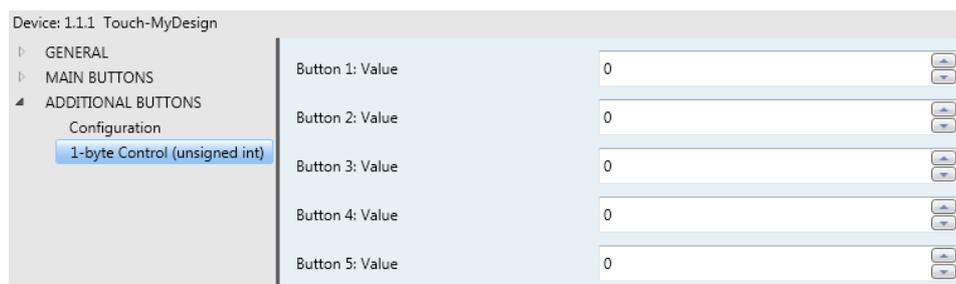


Figure 17 1-byte control: unsigned int (additional buttons)

This type of functionality will always entail a “**state-dependent**” behaviour in the LEDs: as soon as the “[Z] 1-byte value (unsigned int)” object receives a value that equals one of the parameterized values for the additional buttons, the LED corresponding to the matching button will turn on, while the other LEDs will turn off.

### 3.4.4 1-BYTE CONTROL (SCALING)

This option permits parameterizing the five additional buttons all together in order to implement a joint 1-byte scaling control so that, on the event of one of them being touched, the KNX bus will be sent (through the “[Z] 1-byte value (scaling)” object) a certain percentage value (0%-100%), depending on the particular button being touched.

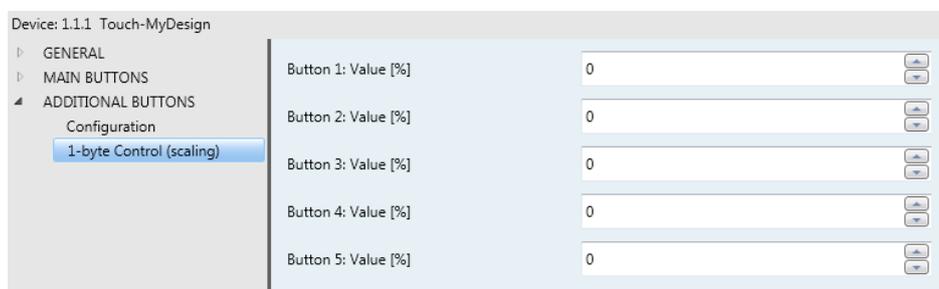


Figure 18 1-byte control: scaling (additional buttons)

This type of functionality will always entail a “state-dependent” behaviour in the LEDs: as soon as the “[Z] 1-byte value (scaling)” object receives a value that equals one of the parameterized values for the additional buttons, the LED corresponding to the matching button will turn on, while the other LEDs will turn.

### 3.4.5 INDIVIDUAL BUTTONS

This option permits using the additional buttons (all or some of them) as individual touch buttons (referred to as T1, T2, T3, T4 and T5), i.e., as touch buttons with independent functionalities, separately configurable. This brings the possibility of making use of up to 5 more individual buttons apart from those from the main button panel, or even destining the latter for pair button controls while the required individual controls can be implemented by the additional button panel.

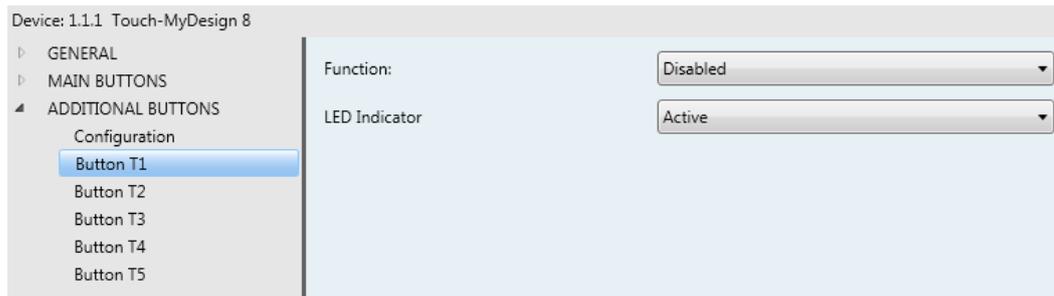


Figure 19 Button T1 (additional buttons)

Therefore, the parameterization and the functionalities available for these additional buttons working as individual controls are completely analogous to those already described for the main buttons working as individual controls. Please refer to section 3.3.2 (omitting, if desired, the two initial paragraphs prior to Figure 14) for further information, although note that the following remarks apply:

- Being the main button panel organised into pairs (A, B, C and D, in the case of Touch-MyDesign 8), the main buttons are referred to as A1, A2, B1, B2, etc. In the case of the additional button panel, the 5 buttons constitute a sole block (“Z”), because of which the additional individual touch buttons will be referred to as Z1, Z2, Z3, Z4 and Z5.
- As a consequence of the above, the name of any object related to an additional button will always begin with “[Zi]”, where “i” is the number of the corresponding button (1-5). Section 3.3.2 refers to object names beginning with “[Xi]”.

### 3.4.6 INDIVIDUAL INDICATORS

This option permits turning the additional button panel non-functional (thus performing no actions on button touches) but making use of the corresponding LEDs as individual indicators whose light statuses permanently depend on the value of the “[Zi] LED On/Off” binary objects, displayed for that purpose.

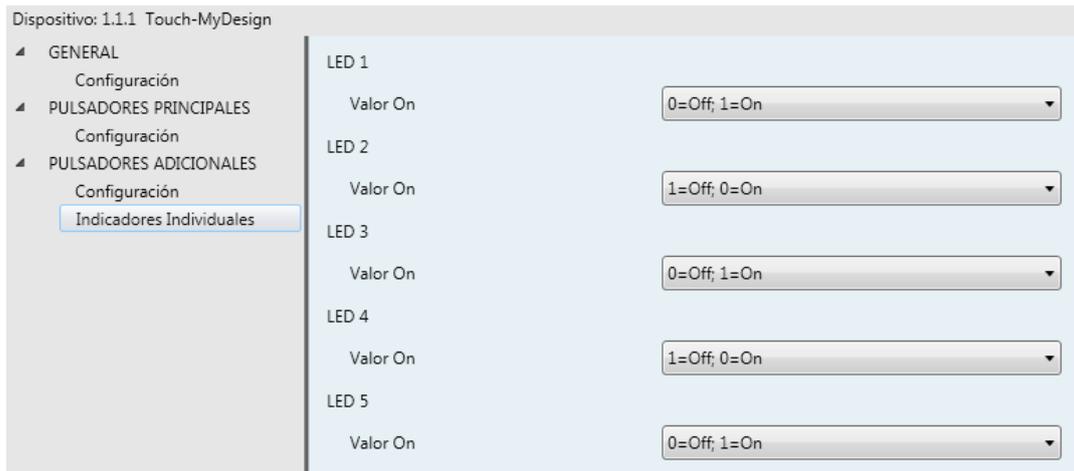


Figure 20 Individual indicators (additional buttons)

One parameter named “ON Value” will be shown for each of the five LEDs, making it possible to associate a certain logic state (“0” ó “1”) to each light state of the LEDs (on and off). The options available for this parameter are:

- **0=Off; 1=On:** the LED will switch to the “on” state (default or customized; see “LED Lighting” in Section 3.2) when the “[Zi] LED On/Off” object receives the value “1”, and will switch to the “off” state when it receives the value “0”.
- **1=Off; 0=On:** the LED will switch to the “on” state (default or customized; see LED lighting in Section 3.2) when the “[Zi] LED On/Off” object receives the value “0”, and will switch to the “off” state (default or customized) when it receives the value “1”.

It is possible to link (via group addresses) the “[Zi] LED On/Off” objects to objects from other devices in the domotic environment, thus making the LEDs of the additional button panel act as indicators of the current state of those objects at any time.

### 3.5 INPUTS

The “Inputs” tab, displayed upon the enablement of such function from General > Configuration, permits setting the required parameters for using up to two input devices (push buttons, switches/sensors, temperature probes or movement detectors) connected to Touch-MyDesign through the corresponding input clamp.



Figure 21 Configuration (inputs)

Within the “Inputs” tab, the “Configuration” window will be available by default, from where it is possible to select the number of inputs to be configured as well as their types, after which two more tabs will be displayed, in order to parameterize the behaviour of the selected inputs.

### 3.5.1 PUSH BUTTON

Configuring an input as a push button will require the definition of the actions to be performed on the event of button presses (both short and/or long).

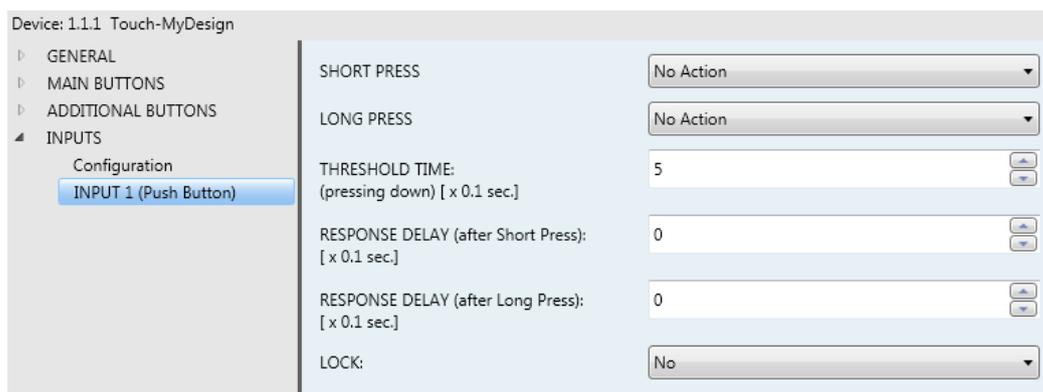


Figure 22 Push button (inputs)

The available options are:

- **Short press:** permits selecting an action to be performed when a short press takes place:
  - **No action:** no action will be performed.
  - **1-bit generic control:** on button presses the KNX bus will be sent the “[Ix] [Short press] 1-bit generic control: B” 1-bit communication object with value B, where B (“0”, “1” or a value that commutes with every press) as well as the possibility of making this sending periodical can be parameterized from the new screen displayed by ETS when required (see Figure Figure 23). For

its part, the periodicity may be assigned to the sending of the values “0” or “1” (or both), and adopt cycle times of 0 to 255 seconds.

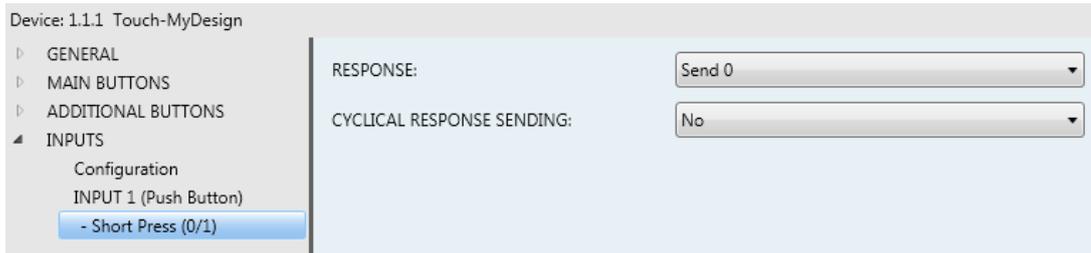


Figure 23 Short press: 0/1 (inputs > push button)

- **Shutter control:** on button presses the KNX bus will be sent a certain shutter control order through the “[Ix] [Short press] ACTION”, where “ACTION” will depend on the action selected from the corresponding window displayed by ETS:



Figure 24 Short press: shutter (inputs > push button)

The available actions are: Move shutter up (the value “0” will be sent), Move shutter down (the value “1” will be sent), Move shutter (toggle direction) (after the first press the value “1” will be sent, while after the second press the value “0” will be; and so on after any subsequent press, commuting the value every time), Stop / Step up (the value “0” will be sent), Stop / step down (the value “1” will be sent) and Stop / step (toggle direction) (after the first press the value “1” will be sent, while after the second press the value “0” will be; and so on after any subsequent press, commuting the value every time).

- **Light dimming:** the KNX bus will be sent a light dimming order through the “[Ix] [Short press] ACTION”, where “ACTION” will depend on the action selected from the corresponding screen displayed by ETS:



Figure 25 Short press: light dimming (inputs > push button)

The available actions are: Light OFF (the value “0” will be sent), Light ON (the value “1” will be sent), Light ON/OFF (Toggle) (after the first press the value “1” will be sent; after the second press, the value “0” will be; and so on after any subsequent presses, commuting the value every time), Decrease light / Stop dimming (Toggle) (after the first press the bus will be sent an order to decrease the light level according to the parameterized light step: 100%, 50%, 25%, 12.5%, 6.25%, 3.1% or 1.5% (see Table 2), while after the second press, the value “0” will be sent; and so on after any subsequent presses, commuting the value every time), Increase light / Stop dimming (Toggle) (after the first press the bus will be sent an order to increase the light level according to the parameterized light step: 100%, 50%, 25%, 12.5%, 6.25%, 3.1% or 1.5% (see Table 2), while after the second press, the value “8” will be sent; and so on after any subsequent presses, commuting the value every time) and Light dimming (toggle) (a combination of the two previous cases, according to the sequence increase light → stop → decrease light → stop).

- **Sending of a scene:** the KNX bus will be sent a certain scene value through the “[Ix] [Short Press] ACTION” object, where “ACTION” will depend on the action configured from the corresponding screen displayed by ETS:

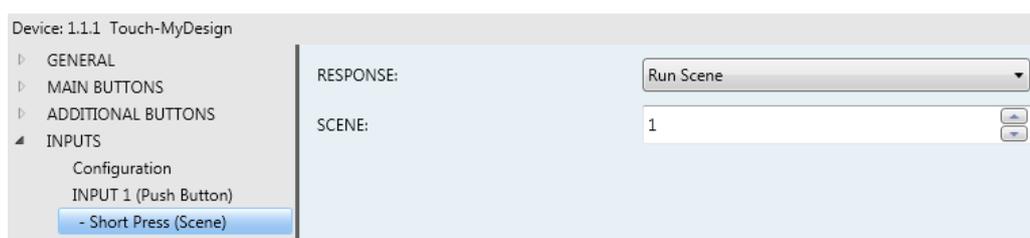


Figure 26 Short press: scene (inputs > push button)

The available actions are:

**Run scene:** the bus will be sent the order to run a scene, i.e., the scene number set for the “Scene” parameter will be sent, decreased by 1 according to the KNX standard, so that other devices linked to the same object execute the corresponding scene (predefined joint reaction).

**Save scene:** the bus will be sent the order to run a scene, i.e., the scene number set for the “Scene” parameter will be sent, increased by 127 according to the KNX standard, so that other devices linked to the same object replace the reaction previously defined for such scene number.

- **Long press:** permits selecting the reaction when a long press takes place. It is completely analogous to the above “Short press” parameter, so please refer to its description for further information.
- **Threshold time:** defines the minimum time (in tenths of a second) that the push button connected to the input of Touch-MyDesign should remain hold before the devices understands it as a long press.
- **Response delay (after short press):** defines a delay (in tenths of a second) to be applied prior to sending the communication object corresponding to the short presses. In other words, when a short press takes place, Touch-MyDesign will wait for the time set for this parameter before sending the KNX bus the value of the corresponding object. To obtain an immediate sending (with no delay), the value “0” may be set here.
- **Response delay (after long press):** analogous to the previous parameter, but referred to long presses.
- **Lock:** selecting “Yes” will display a new 1-bit communication object, “[X] Lock”, that permits locking or unlocking the corresponding input (by sending the values “1” or “0”, respectively). While the input remains locked, any order it sends will be ignored.

### 3.5.2 SWITCH/SENSOR

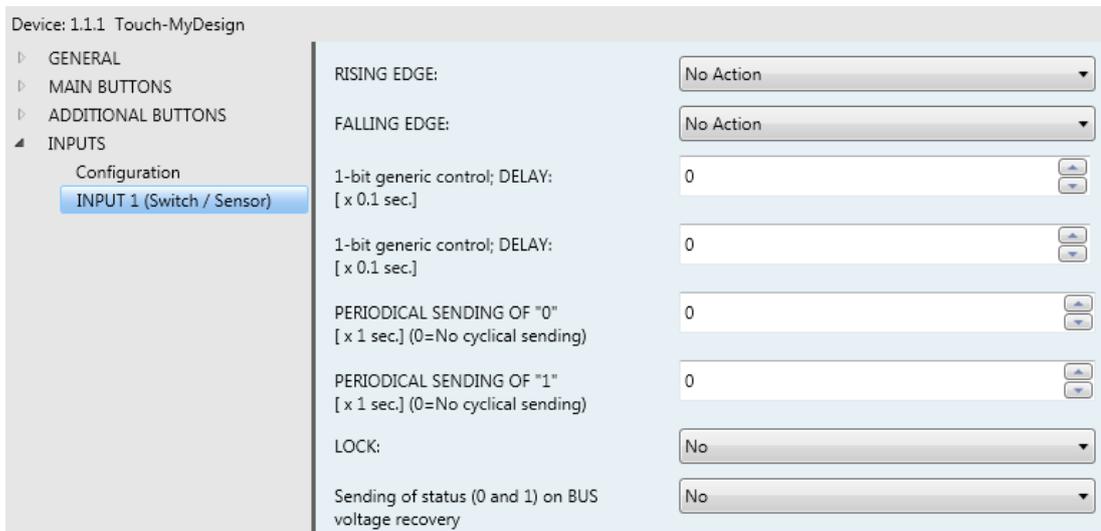


Figure 27 Switch/sensor (inputs)

Configuring an input as a switch/sensor will require the definition of the actions to be performed (all of them through the “[Ix] Edge: binary control” object) on the event of a change in the logical level of the input line. A new screen is displayed for that purpose.

The available options are:

- **Rising edge:** selects the action triggered by a rising edge in the line (which takes place whenever the switch closes the circuit):

**No action:** no action is triggered.

**Send 0:** the value “0” is sent through the “[Ix] Edge: binary control” object.

**Send 1:** the value “1” is sent through the “[Ix] Edge: binary control” object.

**Toggle 0/1:** after the first press, the value “1” will be sent through the “[Ix] Edge: binary control” object; after the second press, the value “0” will be sent. And so on with the subsequent presses, commuting the value every time.

- **Falling edge:** selects the action triggered by a falling edge in the line (which takes place whenever the switch opens the circuit). The available options are analogous to those in the “Rising edge” parameter.

- **Sending of “0”, DELAY:** defines a delay (in tenths of a second) to be applied prior to sending the value “0”. The time count starts when the actual rising/falling edge that originates the sending takes place.
- **Sending of “1”, DELAY:** defines a delay (in tenths of a second) to be applied prior to sending the value “1”. The time count starts when the actual rising/falling edge that originates the sending takes place.
- **Periodical sending of “0”:** defines a cycle time, in seconds, for cyclically re-sending the value “0”. If this parameter is set to “0”, no cyclical re-sending will take place for “0”.
- **Periodical sending of “1”:** defines a cycle time, in seconds, for cyclically re-sending the value “1”. If this parameter is set to “0”, no cyclical re-sending will take place for “1”.
- **Lock:** selecting “Yes” will display a new 1-bit communication object, “[Ix] Lock”, that permits locking or unlocking the corresponding input (by sending the values “1” or “0”, respectively). While the input remains locked, any order it sends will be ignored.
- **Sending of status (0 and 1) on bus voltage recovery:** when this option is enabled, the device will perform an automatic sending of the status of the input line through the “[Ix] Edge: binary control” object whenever a bus power recovery happens. A certain delay (0-255 seconds) will apply if a value different to “0” is set for the “Sending delay” parameter.

### 3.5.3 TEMPERATURE PROBE

---

Configuring an input as a temperature probe will bring up the objects “[Ix] Current temperature” (2 bytes) and “[Ix] Probe error” (1 bit) to the Topology window in ETS. The first one reflects the current value of the temperature measured by the probe connected to the input clamp of Touch-MyDesign. The second object, for its part, will inform about whether there is an error in the connection of the probe (if so, the value of the object will be “1”) that is preventing the reception of the measured values.

Moreover, when an input has been configured as a temperature probe, it will be possible to set a few more parameters from the window that is displayed for that purpose.

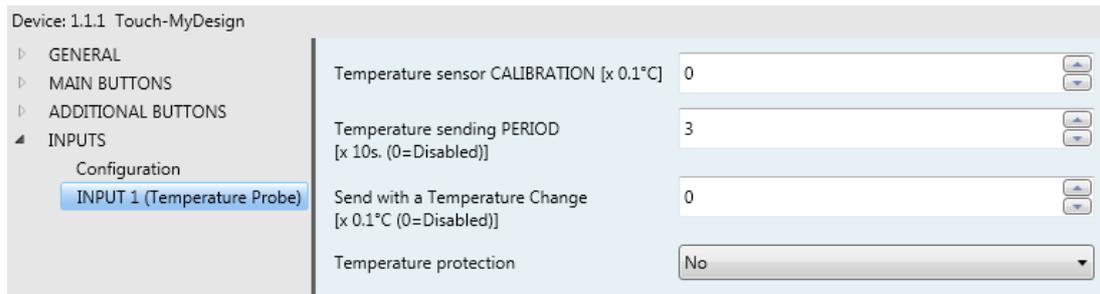


Figure 28 Temperature probe (inputs)

The available options are:

- **Temperature sensor calibration:** defines a certain value (between -50 and +50 tenths of a °C) that will be added to the value received from the probe, so that measurements can be calibrated and deviations can be corrected.
- **Temperature sending period:** defines a cycle time, in seconds, for cyclically re-sending to the KNX bus (through “[Ix] Current temperature”) the updated value measured by the probe. If this parameter is set to “0”, no cyclical re-sending will take place.
- **Send with a temperature change:** defines a relative margin (between 0 and 200 tenths of a degree) in the temperature, so when the difference between two consecutive measurements is greater than such margin, the bus will be automatically sent the newest value (through the “[Ix] Current temperature” object), even if no periodical sending has been parameterized.
- **Temperature protection:** permits activating temperature protection (for overheating, for overcooling or for both). Depending on the selected type or protection, one or two binary communication objects (“[Ix] Overheating” and “[Ix] Overcooling”) will appear with the aim of reflecting (with the value “1”) whether the limit corresponding to each case has been reached or not. This will require the definition of the overheating and overcooling limit temperatures (in degrees), as well as a certain hysteresis (in tenths of a degree) with the intention of avoiding that these objects are re-sent over and over when the temperature becomes almost stable around the limit (exceeding it multiple times).

### 3.5.4 MOVEMENT DETECTOR

Touch-MyDesign permits the connection of movement detectors to its input, each of which provides up to two detection channels.

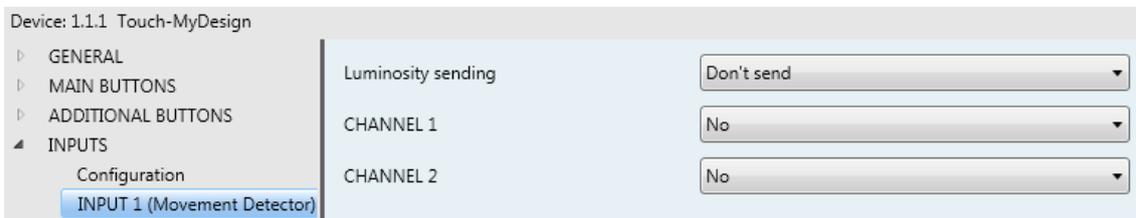


Figure 29 Movement detector (inputs)

Enabling each of the two available channels will bring up a new parameter window, as in Figure 30.

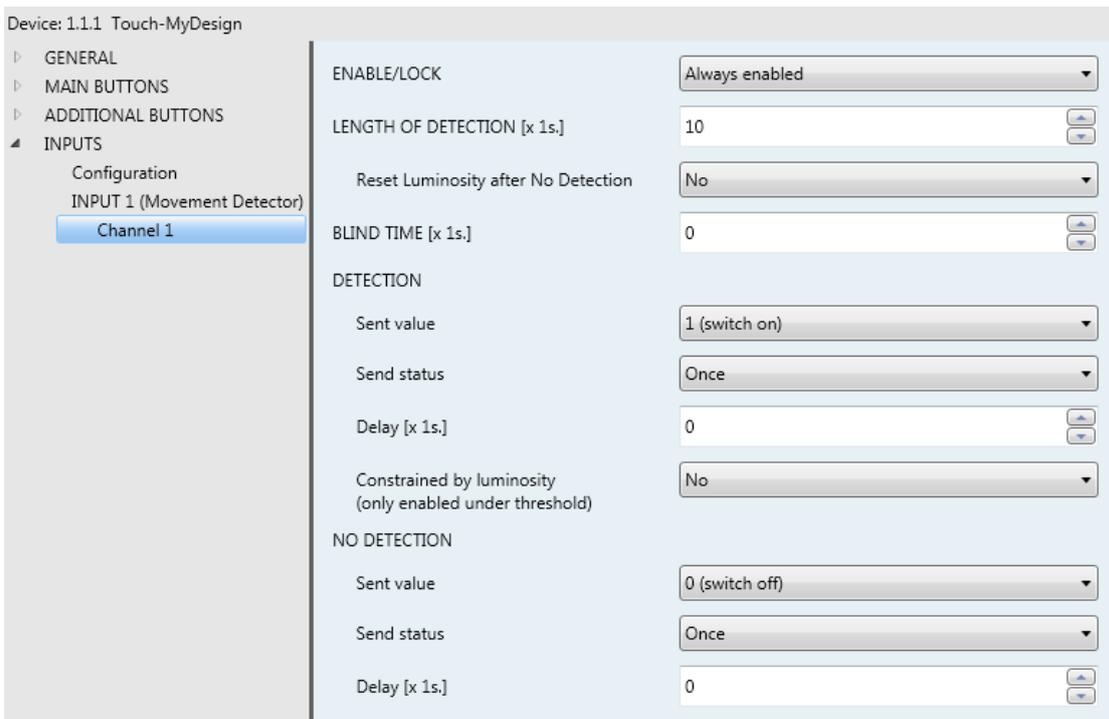


Figure 30 Channel 1 (inputs – movement detector)

For detailed information about the behaviour and parameterization of the movement detector, please refer to the specific document “**Movement detector**”, available at <http://www.zennio.com>.

## 3.6 THERMOSTAT

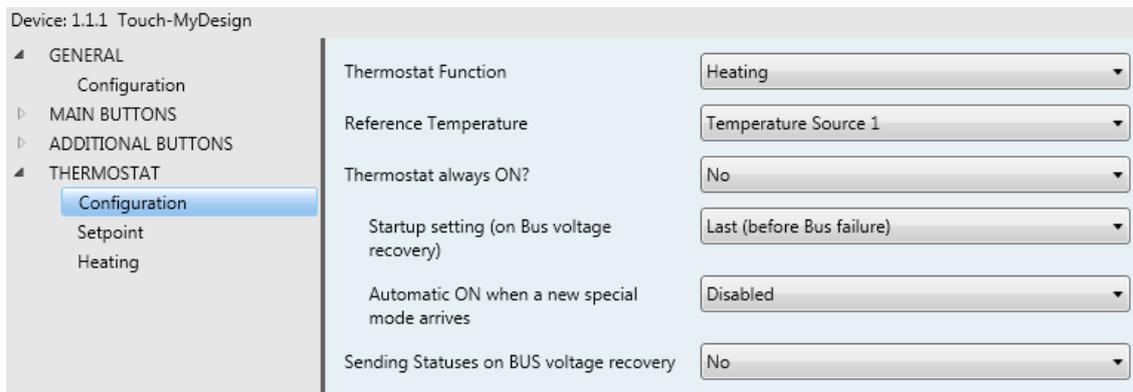


Figure 31 Configuration (Thermostat)

Touch-MyDesign features one thermostat, which can be enabled from the General tab. Once this function has been enabled, a set of additional tabs is displayed (among them, the Configuration window, as shown by Figure 31) so that it can be parameterized.

For detailed information about the behaviour and parameterization of the “Building” thermostat from Zennio, please refer to the specific document “**Zennio Thermostat. ‘Building’**”, available at <http://www.zennio.com>.

# ANNEX I: COMMUNICATION OBJECTS

**IMPORTANT:** this is the list of the communication objects as in their initial state. The name, type and function of some of them may change upon user parameterization.

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
MAIN BUTTONS	0-1	1 bit	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	2	4 bits	O	RT	0-15	0	Last	[A1] Light Dimming (Toggle)	4-bit dimming control
	3-5	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	6	4 bits	O	RT	0-15	0	Last	[A2] Light Dimming (Toggle)	4-bit dimming control
	7-9	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	10	4 bits	O	RT	0-15	0	Last	[B1] Light Dimming (Toggle)	4-bit dimming control
	11-13	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	14	4 bits	O	RT	0-15	0	Last	[B2] Light Dimming (Toggle)	4-bit dimming control
	15-17	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	18	4 bits	O	RT	0-15	0	Last	[C1] Light Dimming (Toggle)	4-bit dimming control
	19-21	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	22	4 bits	O	RT	0-15	0	Last	[C2] Light Dimming (Toggle)	4-bit dimming control
	23-25	2 Bytes	O	RT	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	26	4 bits	O	RT	0-15	0	Last	[D1] Light Dimming (Toggle)	4-bit dimming control

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
MAIN BUTTONS	27-29	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	30	4 bits	O	R T	0-15	0	Last	[D2] Light Dimming (Toggle)	4-bit dimming control
ADDITIONAL BUTTONS	31-33	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	34	4 bits	O	R T	0-15	0	Last	[Z1] Light Dimming (Toggle)	4-bit dimming control
	35-37	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	38	4 bits	O	R T	0-15	0	Last	[Z2] Light Dimming (Toggle)	4-bit dimming control
	39-41	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	42	4 bits	O	R T	0-15	0	Last	[Z3] Light Dimming (Toggle)	4-bit dimming control
	43-45	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	46	4 bits	O	R T	0-15	0	Last	[Z4] Light Dimming (Toggle)	4-bit dimming control
	47-49	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	50	4 bits	O	R T	0-15	0	Last	[Z5] Light Dimming (Toggle)	4-bit dimming control
GENERAL	51	2 Bytes	O	R T	Ac. to param.	Ac. to param.	Ac. to param.	[Accord. to param.]	
	52	1 Byte	I	W	0-63	Irrelevant	Irrelevant	[General] Scene: receive	0-63 (Run Scene 1-64)
	53	1 Byte	O	T	0-63, 128-191	Irrelevant	Irrelevant	[General] Scene: send	0-63 (Run or Save Scene 1-64)
	54	1 bit	I	W	0/1	1	Last	[General] LEDs: normal mode	1=Normal Mode; 0=No Action
	55	1 bit	I	W	0/1	0	Last	[General] LEDs: night mode	1=Night Mode; 0=No Action

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
GENERAL	56	1 bit	I	W	0/1	Irrelevant	Last	[General] Touch Lock	1=Lock; 0=Nothing
	57	1 bit	I	W	0/1	Irrelevant	Last	[General] Touch Unlock	1=Unlock; 0=Nothing
	58	1 bit	I	T	0/1	Irrelevant	Irrelevant	[General] Welcome Back Object	1-bit generic control
	59	1 bit	I	W	0/1	0	Last	[General] Flash LEDs	1=Flash LEDs; 0=No Action
	60	1 bit	I	W	0/1	0	Last	[General] Buzzer	1=Beep; 0=Nothing
	61	1 bit	I	W	0/1	1	Last	[General] Buzzer Enabling	1=Enabled; 0=Disabled
THERMOSTAT	62	2 bytes	I	W	-40°C – 150°C	25°C	Last	[T] Temperature Source 1	External Sensor Measure
	63	2 bytes	I	W	-40°C – 150°C	25°C	Last	[T] Temperature Source 2	External Sensor Measure
	64	1 byte	I	W	1-4	Ac. to param.	Ac. to param.	[T] Special Mode	1-byte HVAC Mode
	65	1 bit	I	W	0/1	0	Last	[T] Special Mode: comfort	0=Off; 1=On
	66	1 bit	I	W	0/1	0	Last	[T] Special Mode: standby	0=Off; 1=On
	67	1 bit	I	W	0/1	0	Last	[T] Special Mode: economy	0=Off; 1=On
	68	1 bit	I	W	0/1	0	Last	[T] Special Mode: protection	0=Off; 1=On
	69	1 bit	I	W	0/1	0	Last	[T] Window Status (input)	0=Closed; 1=Open
	70	1 bit	I	W	0/1	0	0	[T] Comfort Prolongation	0=Nothing; 1=Timed Comfort
	71	1 Byte	O	T R	1-4	Ac. to param.	Last	[T] Special Mode Status	1-byte HVAC Mode

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
THERMOSTAT	72	2 Bytes	I	W	-20°C – 150°C	Ac. to param.	Last	[T] Setpoint	Thermostat setpoint input
	73	1 bit	I	W	0/1	0	Irrelevant	[T] Setpoint Step	0=-0.5°C; 1=+0.5°C
	74	2 Bytes	I	W	-10°C, 10°C	0	Last	[T] Setpoint Offset	Float offset value
	75	2 Bytes	O	TR	-20°C – 150°C	25°C	Last	[T] Setpoint Status	Current setpoint
	76	2 Bytes	O	TR	-20°C – 150°C	Ac. to param.	Last	[T] Basic Setpoint Status	Current basic setpoint
	77	2 Bytes	O	TR	-10°C, 10°C	0	Last	[T] Setpoint Offset Status	Current setpoint offset
	78	1 bit	I	W	0/1	0	Last	[T] Setpoint Reset	Reset setpoint to default
	79	1 bit	I	W	0/1	0	Irrelevant	[T] Mode	0 = Cool; 1 = Heat
	80	1 bit	O	TR	0/1	Ac. to param.	Last	[T] Mode Status	0 = Cool; 1 = Heat
	81	1 bit	I	W	0/1	Ac. to param.	Ac. to param.	[T] On/Off	0=Off; 1=On
	82	1 bit	O	TR	0/1	Ac. to param.	Ac. to param.	[T] On/Off Status	0=Off; 1=On
	83	1 bit	O	TR	0/1	0	Last	[T] Control Variable (Cool)	2-Point Control
	84	1 bit	O	TR	0/1	0	Last	[T] Control Variable (Heat)	2-Point Control
	85	1 Byte	O	TR	0%-100%	0%	Last	[T] Control Variable (Cool)	PI Control (Continuous)
	86	1 Byte	O	TR	0%-100%	0%	Last	[T] Control Variable (Heat)	PI Control (Continuous)
	87	1 bit	O	TR	0/1	0	Last	[T] Additional Cool	Temp >= (Setpoint+Band) => "1"
	88	1 bit	O	TR	0/1	0	Last	[T] Additional Heat	Temp <= (Setpoint-Band) => "1"

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
INPUTS	89-90	1 bit	I	W	0/1	0	Last	[Ix] Lock	1=Input Disabled; 0=Input Free
	91-92	1 bit	I	TW	0/1	0	0	[Ix] [Edge] Binary Control	1-bit generic control
	93-94	4 bits	O	TR	0-15	0	Last	[Ix] [Short Press] Inc. Light / Stop Dim. (Toggle)	Increase Light / Stop Dimming (Toggle)
	95-96	1 Byte	O	TR	128-191	Irrelevant	Irrelevant	[Ix] [Short Press] Save Scene	Save Scene -> Send. of 128-191
	97-98	1 bit	I/O	TRW	0/1	0	Last	[Ix] [Long Press] Binary Control: "0"	1-bit generic control;
	99-100	4 bits	O	TR	0-15	0	Last	[Ix] [Long Press] Inc. Light / Stop Dim. (Toggle)	Increase Light / Stop Dimming (Toggle)
	101-102	1 Byte	O	TR	128-191	Irrelevant	Irrelevant	[Ix] [Long Press] Save Scene	Save Scene -> Send. of 128-191
	103-104	2 Bytes	O	TR	-20-95°C	25°C	Last	[Ix] Current Temperature	Temperature sensor value
	106-107	1 bit	O	TR	0/1	0	Last	[Ix] Overcooling	1=Overcooling;0=No Overcooling
	109-110	1 bit	O	TR	0/1	0	Last	[Ix] Overheating	1=Overheating;0=No Overheating
	112-113	1 bit	O	TR	0/1	Ac. to status	Ac. to status	[Ix] Probe Error	1=Error;0=No Error
	114-115	1 bit	O	TR	0/1	0	0	[Ix] Short Circuit	1=Short Circuit;0=No Short Cir
	116-117	1 bit	O	TR	0/1	0	0	[Ix] Open Circuit	1=Open Circuit;0=No Open Circ.
	118-119	1 Byte	O	TR	0-100%	Irrelevant	Last	[Ix] Luminosity Level	Luminosity on Input x
	120-123	1 bit	I	W	0/1	0	0	[Ix][Ch.y] Channel enabling	1=Enable; 0=Disable
	124-127	1 bit	O	T	0/1	Ac. to param	Ac. to param.	[Ix][Ch.y] Detection Status	According to parameters
128-131	1 Byte	I	W	0-63	Irrelevant	Irrelevant	[Ix][Ch.y] Scene Reception	0-63 (Run Scene 1-64)	

SECTION	NUMBER	SIZE	IN/OUT	FLAGS	VALUES			NAME	DESCRIPTION
					RANGE	1st TIME	RESET		
INPUTS	132-135	1 Byte	O	T	0-63, 128-191	Irrelevant	Irrelevant	[Ix][Ch.y] Scene Sending	0-63 (Send Scene 1-64)
INTERNAL SENSOR	105	2 Bytes	O	TR	-20-95°C	25°C	Last	[Internal Sensor] Current Temperature	Temperature sensor value
	108	1 bit	O	TR	0/1	0	Last	[Internal Sensor] Overcooling	1=Overcooling;0=No Overcooling
	111	1 bit	O	TR	0/1	0	Last	[Internal Sensor] Overheating	1=Overheating;0=No Overheating

Join and send us your inquiries  
about Zennio devices:

<http://zennioenglish.zendesk.com>

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

*Tel. +34 925 232 002.*  
*Fax. +34 925 337 310.*  
*www.zennio.com*  
*info@zennio.com*



RoHS