



KNX Capacitive Colour Touch Panel

ZN1VI-TP41C

Application Program Version: [2.1]
User Manual Version: [2.1]_a

www.zennio.com

DOCUMENT UPDATES

| Version | Changes | Page(s) |
|--|---|---------|
| 2.1_a | <p>Changes in the application program:</p> <ul style="list-style-type: none"> • Parameter added for setting the attenuation of the backlight (total, partial or none) on inactivity. • Support added for TrueType fonts. • Compatibility with Greek and Cyrillic characters. • Driver update to offer compatibility with newer capacitive touch panels. • Minor changes in the handling of graphics, themes and transitions between themes. • “Lock” icon added to the Profile and Configuration pages when security protection has been set to them. • Interruption of the inactivity timeout counter if a pop-up windows is active. • Unification of the text format and time spacing of labels shown in graphical value monitoring. | - |
| | Note added about the start-up calibration. | 10 |
| | Clarification about the maintenance of the time and date after a restart, and about the hour range (0-23). | 15, 16 |
| | Note added about the interruption of the inactivity time-out counter while pop-up windows are active. | 18 |
| | New parameter about the screen attenuation. | 17, 22 |
| | Note added about text width limitations. | 19 |
| | New name for the scene object. | 20 |
| | Clarification about how the number of decimal figures shown in floating point indicators is dynamically determined depending on the order of magnitude of the current value. | 38 |
| Correction of the range and type of certain objects with types DPT_Value_2_Count y DPT_Value_Tempd | 77-79 | |

| | | |
|--------------|--|----------|
| <p>2.0_a</p> | <p>Changes in the application program:</p> <p>General improvement of the capacities and functionality. Changes in the user interface and in the page browsing.</p> <p>More versatility:</p> <ul style="list-style-type: none"> • 6 general-purpose pages and 2 specific pages. • Possibility of implementing different controls (climate, scenes, etc.) within the same page. • Self-alignment of the boxes in a page (optional). <p>New functionality:</p> <ul style="list-style-type: none"> • RGB control. • RGBW control. • Energetic consumption monitoring. • Daily timer. • Alarm log. • Numerical indicator log. | <p>-</p> |
|--------------|--|----------|

Contents

| | |
|---|----|
| Document updates | 2 |
| 1 Introduction | 6 |
| 1.1 InZennio Z41..... | 6 |
| 1.2 Z41 2.x..... | 8 |
| 1.3 Installation | 9 |
| 1.3.1 Firmware updates..... | 11 |
| 2 Configuration | 12 |
| 2.1 Menu page | 12 |
| 2.2 General-purpose pages..... | 13 |
| 2.3 Profile page | 13 |
| 2.4 Configuration page | 15 |
| 2.5 Backlight..... | 17 |
| 2.6 Page shaping | 18 |
| 3 ETS Parameterization | 19 |
| 3.1 Default configuration..... | 19 |
| 3.2 Main configuration | 21 |
| 3.2.1 General..... | 22 |
| 3.2.2 Security | 24 |
| 3.2.3 Touch lock | 27 |
| 3.2.4 Internal temperature sensor | 28 |
| 3.2.5 Ethernet | 29 |
| 3.2.6 Energy monitor objects..... | 30 |
| 3.2.7 White channel objects | 31 |
| 3.3 Menu..... | 31 |
| 3.3.1 Configuration..... | 32 |
| 3.4 Page <i>n</i> | 33 |
| 3.4.1 Configuration..... | 33 |
| 3.4.2 Box <i>i</i> | 34 |
| 3.4.2.1 Indicator..... | 35 |
| 3.4.2.2 1-button control | 39 |
| 3.4.2.3 2-button control | 42 |

- 3.4.2.4 Climate control.....52
- 3.4.2.5 Other.....60
- 3.5 Profile page72
 - 3.5.1 Configuration.....73
- 3.6 Configuration page73
 - 3.6.1 Configuration.....73
- 3.7 Thermostat n.....74
- ANNEX. Communication Objects.....76

1 INTRODUCTION

1.1 INZENNIO Z41

InZennio Z41 is an easily and intuitively controllable high-performance **colour touch screen**. Its built-in features and functions make it the ideal solution for integral room control in hotels, offices or any other environments where controlling climate systems, lighting systems, shutters, scenes, etc. is required.

The most outstanding features of InZennio Z41 are:

- **4.1 inch** backlit **capacitive** touch panel.
- User interface with various pre-set **colour patterns** and themes.
- Multiple **direct action functions**, fully customizable.
- Control distribution across up to **6 customizable pages**.
- Full **climate** management.
- Programmable **timers**.
- **Scene** control.
- **Alarm** control.
- 2 independent **thermostats**.
- Built-in temperature sensor.
- **Ethernet** interface for firmware updates and for future functionality.
- **Elegant** design, available in various **colours**.

Z41 is offered with two alternate application programs:

- **Z41 1.x**.
- **Z41 2.x**.

Table 1 shows their most significant functionalities and their differences.

| Function | Z41 1.x | Z41 2.x |
|--|---------|---------|
| General-purpose pages | 2 | 6 (***) |
| Specific-purpose pages (climate, scenes, timers) | 4 | |
| Binary indicators (text / icon) | | |
| Enumerated indicators (text / icon) | | |
| Numerical indicators (percentage, floating point, etc.) | (**) | |
| Chronological data log for numerical indicators | | |
| 1-button controls (binary, scene, percentage, floating point, etc.) | (**) | |
| 2-button controls (binary, scene, percentage, floating point, etc.) | (**) | |
| Climate controls | | |
| RGB controls | | |
| RGBW controls | | |
| Energy consumption monitoring | | |
| Daily timers | | 48 (*) |
| Weekly timers | 16 | 48 (*) |
| Alarms | 8 | 48 (*) |
| Alarm log | | |

Table 1 Differences between Z41 1.x and Z41 2.x

(*) As a maximum (making use of all boxes from all pages).

(**) Only positive numeric values.

(***) This differentiation does not apply anymore; six pages are now provided combining all the functionalities together.

Note: the following sections of this user manual will be focused on the **Z41 2.x** application program.

1.2 Z41 2.x

The Z41 2.0 application program features the following functions:

- **6 general-purpose pages, with up to 8 fully-customizable boxes each,** which the integrator may configure as indicators or controls.

- **Indicators:**

- Binary (icon or text).
- Enumerated (icon or text).
- Unsigned integer (8 / 16 bits).
- Signed integer (8 / 16 bits).
- Scaling (percentage).
- Floating point (16 bits).

- **1-button controls:**

- Binary (preset value; switch; hold & release; short press, long press...).
- Unsigned integer (8 / 16 bits).
- Signed integer (8 / 16 bits).
- Scaling (percentage).
- Floating point (16 bits).
- Scene (run / save).

- **2-button controls:**

- Binary (icon, text, number).
- Enumerated (icon, text).
- Unsigned integer (8 / 16 bits).
- Signed integer (8 / 16 bits).
- Percentage (Scaling).
- Floating point (16 bits).
- 2 scenes (run / save).
- Shutters.
- Light dimming.

- **Climate specific controls:**

- Temperature setpoint control.
- Mode (cool/heat or extended).
- Special modes.
- Fan.

- **Other controls:**

- RGB.
- RGBW.
- Energetic consumption (4 Bytes).
- Daily timer.
- Weekly timer.
- Alarm.

- **1 Profile page** (optionally), which lets the user modify, directly from the user interface, parameters such as the brightness, contrast and colour pattern of the screen, and the sound of the button press beeps.

- **1 Configuration page** (optionally), which contains the controls for setting the time/date and for calibrating the built-in temperature probe. It also shows the alarm log, the software version and the current IP address.

1.3 INSTALLATION

Figure 1 shows the connection outline of the InZennio Z41 screen:

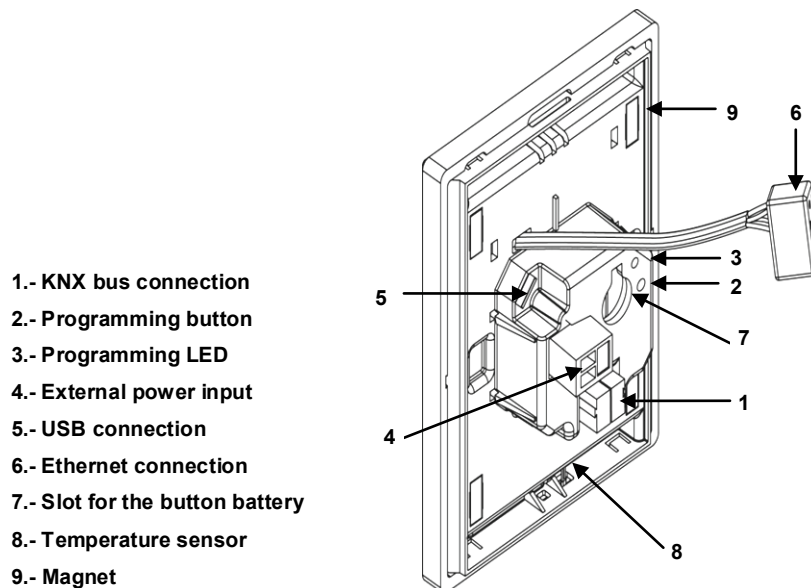


Figure 1 Element diagram

The InZennio Z41 touch screen connects to the KNX bus through the built-in terminal (1). To ensure enough supply as to power all the internal components of the device, it is also necessary to connect an external **DC power supply** of 12, 24 or 29v which delivers at least 150 mA of current. The KNX power and the external power must be electrically isolated and provided by separate supplies, in order to avoid interference to the KNX bus.

This external power must be supplied to the device through the corresponding terminal (4) provided within the original packaging, ensuring that each cable is properly connected and respecting the electrical polarities –positive and negative– from the supply to the device.

In addition to the external supply, Z41 makes use of a **button battery** (LR44), located at the corresponding slot (7), to help maintain the time and date updated in case of a failure of the external power supply.

Note: immediately after connecting the external supply, the device will perform an automatic calibration, which may result deviated if it takes place while any type of pressure is being made over the front panel. In case of an unexpected behaviour in touch recognition, it will be necessary to disconnect and connect the device back, avoiding making pressure over the panel, in order to perform a new calibration.

After the connection of the external power supply, the screen will blink for an instant. Approximately 7 seconds later, a welcome message containing the Zennio logo and the “Loading...” label will be shown. Once the load process ends, the **Status page** will come up (see Figure 2).

| | | |
|---------------|------------|--------|
| 12:45 | 20/06/2012 | 29.7°C |
| Zennio Z41 | | |
| 01.01.001 | | |
| KNX | | |
| OK | | |
| Software | | |
| 0% | | |
| Sync | | |
| 192.168.1.220 | 2.0.0 | |
| ETHERNET | SW Version | |

Figure 2 Status page

This screen is divided into 8 boxes:

- The “**KNX**” box shows the individual address assigned to the screen (in case of no assignment, this will show as “x.x.x”).
- The “**Software**” box will show empty in absence of an application program. In general, it will indicate if the download software is correct (“OK”) or if it is found to be corrupt due to download issues (“ERROR”).
- The “**Sync**” box shows, as a percentage, the progress of the internal synchronisation of the device during the start-up. Once it reaches 100%, the Status page will disappear and the Menu page will come up.
- The “**Ethernet**” box shows the IP address assigned to Z41, provided that an Ethernet cable has been attached (6) and the network is running a DHCP server.

- Last, the “**SW Version**” box shows the version of the application program currently loaded to the device.

The Status page will also show up during bus failures in case the external power (12-24-29v) is not interrupted.

Note: *although the device will not turn functional until it is powered with the external supply, the KNX bus power should be enough to perform downloads from ETS (application program, group addresses, etc.).*

A brief press on the **programming button** (2) will make the device enter the programming mode. The **programming LED** (3) will then light in red. On the contrary, if this button is held while the device gets connected to the bus, Z41 will enter the **secure mode**. In such case, the programming LED will blink in red colour.

Connectors (5) and (6) are provided for firmware updates (see section 1.3.1) and eventually for future functionality.

For detailed information about the technical features of Z41, as well as on security and installation procedures, please refer to the device **Datasheet**, included within the device packaging and also available at <http://www.zennio.com>.

1.3.1 FIRMWARE UPDATES

Z41 incorporates **two different microcontrollers**. One of them is focused on interfacing with the KNX bus and on running the application program itself, while the second one is dedicated to running the firmware that implements the operating system and the management of the peripherals.

Therefore, an eventual update of Z41 may consist in two phases:

- Downloading the new application program (from ETS),
- Downloading the new firmware.

Two alternative methods are provided for the latter: the connection of a flash memory drive to the USB slot (see 6 in Figure 1) through the proper adapter, or the connection of an Ethernet network cable (see 6 in Figure 1). For further details on the firmware update process, please refer to the specific document “**Z41 – USB/Ethernet firmware update**”, available at <http://www.zennio.com>.

2 CONFIGURATION

2.1 MENU PAGE

The user interface of the InZennio Z41 touch screen is organised into **pages** (up to eight different pages), each of which can be accessed from the **Menu page**, which (unless parameterized the contrary) is automatically shown after the start-up of the device.

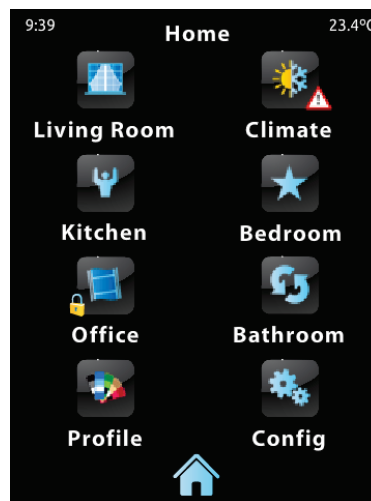


Figure 3 Menu page (example)

As Figure 3 shows, the access to the pages is provided by a set of icons, which eventually may show other (smaller) icons overlaid, such as an **exclamation** or a **lock**. Exclamations reflect that there are active alarms (see epigraph f) in section 3.4.2.5) within the page, while a lock reflects that the access to the page is password-protected (see section 3.2.2).

On the other hand, 6 out of the 8 pages are **general-purpose** pages at the entire disposal of the integrator, who may include up to 8 controls/indicators (with different functionalities) per page, being even possible to combine alarm, climate or any other controls within the same page, in contrast to previous versions of the application program. The remaining two pages (**Profile** and **Configuration**) are specific-purpose, as they are provided for user customization of the device in runtime.

The user interface permanently shows a top bar containing the name of the current page as well as the current time and temperature. A button can be also found at the bottom of every page to get back to the Menu page:



Figure 4 Menu button

2.2 GENERAL-PURPOSE PAGES

The integrator can make use of up to 6 general-purpose pages, each of which can hold up to 8 different indicators or controls (with no restrictions for combining them) which will show inside the pages, automatically distributed or not, according to page parameterization. For more details on the box distribution, refer to section 2.6.

The parameterizable controls for the general-purpose pages and their behaviour will be described in detail in section 3.

2.3 PROFILE PAGE

The Profile page lets the final user perform certain adjustments on the visual and audible side of the user interface:



Figure 5 Profile page

The specific controls that can be made available for the Profile page are:

- **Contrast.**
- **Brightness.**

- **Theme:** desired colour combination, among eight different options:



Figure 6 Themes

- **Melody:** desired tune (among three different options) for the beeps emitted on button presses or as a feedback on action executions. One more possibility is to make these beeps silent (however alarm beeps will still work).

Note that the labels of the controls in the Profile page, as well as the title of the page itself, are customizable in ETS. Moreover, it is possible to make any of the described controls remain hidden for the final user. These details are explained in section 3, together with the entire parameterization process.

2.4 CONFIGURATION PAGE

The Configuration page lets the user know or adjust certain technical values about the device.

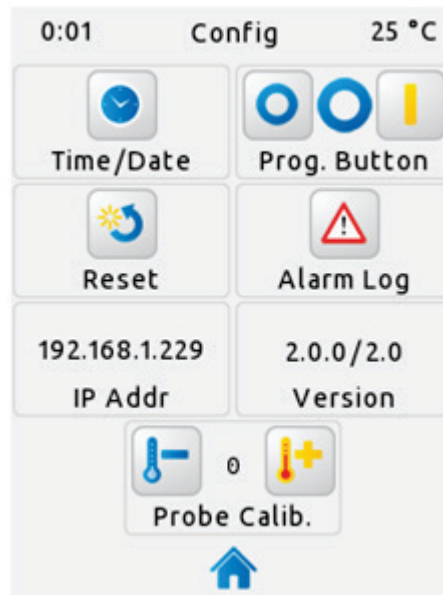


Figure 7 Configuration page

The specific controls that can be made available for the Configuration page are:

- Date / time:** even though the device implements two communication objects (see section 3.1) for setting the internal time and date from an external device, they can also be adjusted (if such option has been enabled by parameter) directly from the user interface. Therefore, by pressing this button, a set of controls will pop up to let the user set the current hour (0-23), minutes, day, month and year.



Figure 8 Time set control

Z41 features an internal clock powered by a battery (see section 1.3), because of which the time/date will not be lost after downloads or reboots.

- Programming button:** control/indicator that shows the status of the programming LED of the device. In particular, it permits entering/leaving the

programming mode as by pressing the actual programming button of the device (see section 1.3) in case the rear of Z41 cannot be accessed.

- **Reset:** a long press (~5 seconds) on this button sets the device back to the state it had after being programmed. It is equivalent to performing a reset from ETS. Nevertheless, data and alarm logs are not set to zero after a reset.

Note: *the reset option causes the restitution of any adjustments made by the user from the Profile and Configuration pages, except the time and date.*

- **Alarm log:** log file where alarm events will be incorporated as soon as they occur. This centralised file may be useful when the user needs to check the status of a set of alarm controls that have been distributed across several pages.

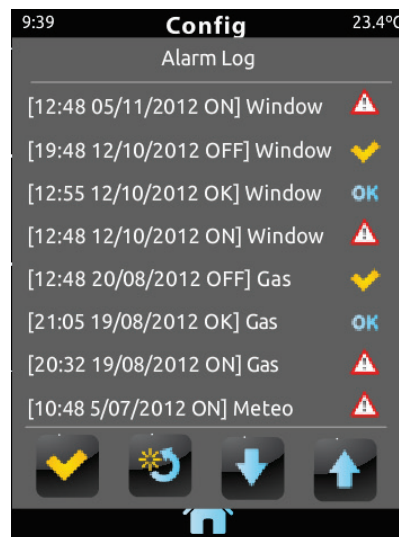


Figura 9 Alarm log

Hence, the alarm log will contain a listing, in reverse chronological order, where the name, time and date of every alarm event will be shown. In addition, the log will show the current status of each of them, which may be:

- **“ON”:** the alarm is still active or has not been acknowledged by the user.
- **“OK”:** the user has acknowledged the alarm event, but the corresponding object is still in the alarm state.
- **“OFF”:** the user has acknowledged the alarm event and the corresponding object has already left the alarm state.

For more information on alarms, see epigraph f) in section 3.4.2.5.

- **IP address:** in case the device has been connected to an IP network through an Ethernet cable (see section 1.3), the IP address it has been assigned (if any) will be shown here.
- **Version:** indicator that informs about both, the version of the application program and, next to that, the version of the firmware that have been downloaded to the device (see section 1.3.1).
- **Probe calibration:** control for setting a certain correction over the values thrown by the internal temperature probe. This way, if the value of the measurement is found to be slightly over (or below) the actual temperature due to external factors, the final user will have the option to set an adjustment of between -5.0°C and $+5.0^{\circ}\text{C}$, in steps of 0.1°C . In addition, it is possible to set an initial value for such correction by parameter.

Note: *any changes made on the calibration of the internal probe will be applied beginning with the next measurement performed. Therefore, they may not be perceived immediately.*

Note that the labels of the controls in the Profile page, as well as the title of the page itself, are customizable in ETS. Moreover, it is possible to make any of the described controls stay hidden for the final user. These details are explained in section 3, together with the entire parameterization process.

2.5 BACKLIGHT

In order to prevent unnecessary power consumptions while the device is idle, Z41 automatically fades partially out the backlight of the screen after two minutes without any user interaction. Next, after three more minutes of inactivity (i.e., after five minutes of inactivity), the backlight is completely turned off. While partially or totally faded out, any press on the screen will bring back the normal light level.

The above timings do not apply while a pop-up window is open or if a particular backlight configuration has been parameterized (see section 3.2.1).

2.6 PAGE SHAPING

Each of the pages of the user interface –including Menu, Profile and Configuration– can be independently parameterized to display its control/indicator boxes statically distributed or automatically ordered.

- **Static (grid) distribution:** all the 8 boxes of the page will be shown, remaining empty those not having been assigned a function.

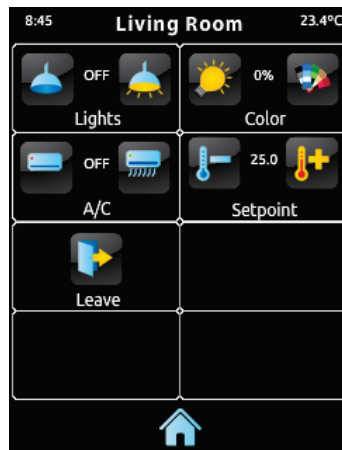


Figure 10 Static distribution

- **Automatic distribution:** the active boxes will be dynamically ordered, so that they make use of all the available space. The final layout of the page will therefore depend on the number of boxes having been assigned a function.

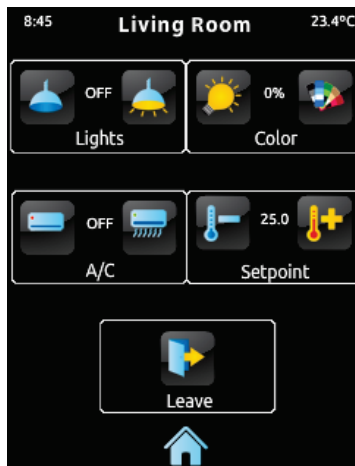


Figure 11 Dynamic distribution of the enabled boxes

3 ETS PARAMETERIZATION

To begin with the parameterization of InZennio Z41, once the ETS program is running it is necessary to import the product database (**Z41** application program). Next, the device is added to the project and, after right-clicking on the name of the device, the option “Edit parameters” must be selected to start the configuration process.

The next sections explain in depth the ETS parameterization of the device.

Note: *the number of characters permitted for text parameters (labels, etc.) may vary depending on whether special characters (symbols, accents...), coded with more than one byte, are being used or not. ETS does not dynamically adjust the width of this parameters, because of which a warning may be shown afterwards stating that the limit has been exceeded, even if the textbox is still showing free spaces.*

Important: *whenever the device is updated to a new version of the application program from ETS, it is necessary to also update the firmware of the secondary microcontroller (responsible for the operating system; see section 1.3.1) in case an updated version exists. Should the installed versions of the application program and the firmware not match each other, the device may not work correctly (eventually making the Software box in the Status page show the label “incompatible”).*

3.1 DEFAULT CONFIGURATION

This section illustrates the default initial state of the device.

| | | | | | | | | | | |
|---|--------------------------------|---------------------------------------|--------|---|---|---|---|---|------------------|-----|
| 0 | [General] Time | Current Time | 3 Byte | C | R | W | T | U | time of day | Low |
| 1 | [General] Date | Current Date | 3 Byte | C | R | W | T | U | date | Low |
| 2 | [General] Scenes | Scene Value | 1 Byte | C | - | - | T | - | scene control | Low |
| 3 | [General] Internal Temperature | Current Temperature (built-in sensor) | 2 Byte | C | R | - | T | - | temperature (°C) | Low |
| 5 | [General] Illumination | 0=No action; 1=Light Display | 1 bit | C | - | W | - | - | acknowledge | Low |
| 6 | [General] Touch Lock | 0=Touch enabled; 1=Touch disabled | 1 bit | C | - | W | - | - | on/off | Low |

Figure 12 Default topology

The general objects enabled by default are:

- **[General] Time:** 3-byte object for externally (for example, by linking it to a KNX clock) setting the internal time of the device. This object also permits read requests, so the current time of the device can be checked. It is also automatically sent after time changes made by the user from the screen itself.

- **[General] Date:** 3-byte object for externally (for example, by linking it to a KNX clock) setting the internal date of the device. This object can also accept read requests, so the current date of the device can be checked. It is also automatically sent after date changes made by the user from the screen itself.
- **[General] Scenes: send:** 1-byte object through which scene run/save orders (1 to 64 according to the KNX standard) can be sent to the bus in case any of the boxes is configured as a scene control or as a timer.
- **[General] Internal temperature:** 2-byte object through which the bus will be sent, periodically or according to the parameterization, the value of the current measurement of the built-in sensor probe (see section 3.2.4).
- **[General] Illumination:** 1-bit object that, upon the reception of the value “1” from the bus, will bring the backlight of the screen to the maximum level, while the reception of the value “0” will have no effect. This maximum illumination will last for a certain time, according to the explanation in section 2.5.
- **[General] Touch lock:** 1-bit object that, upon the reception of the value “1” from the bus, will lock the touch panel so that user presses are not taken into account anymore. Presses will no longer be ignored as soon as the value “0” is received through this object. See section 3.2.3.

On the first parameter edition, a screen similar to Figure 13 will be shown.

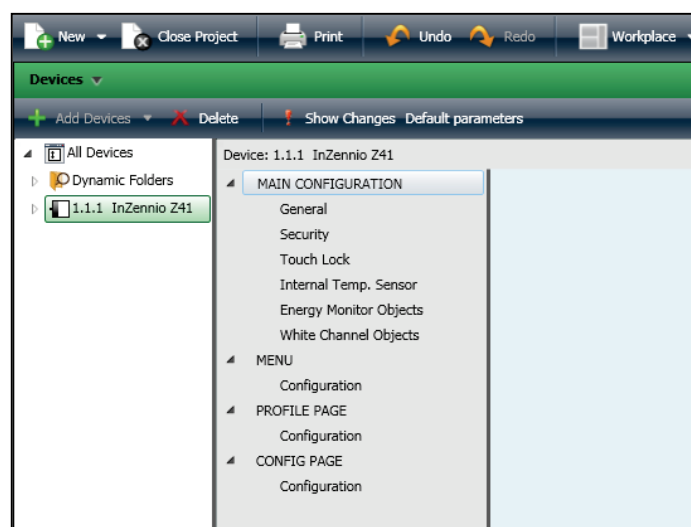


Figure 13 Parameter screen shown by default

As shown in Figure 13, the parameter window is initially divided into four main tabs:

- **Main configuration**, which will contain six more screens by default:
 - General,
 - Security,
 - Touch lock,
 - Internal temperature sensor,
 - Energy monitor objects,
 - White channel objects.

- **Menu**, which will contain one more screen by default:
 - Configuration, from which it will be possible to activate and configure each of the six general-purpose pages of the device (see section 2.2).

- **Profile page**, which itself will contain one more screen:
 - Configuration, from which it will be possible to activate and configure the Profile page (see section 2.3).

- **Config. page**, which itself will contain one more screen:
 - Configuration, from which it will be possible to activate and configure the Configuration page (see section 2.4).

The following sections will detail the parameterization process of each of the above tabs, including the initially hidden screens, which will become visible depending on the selected options.

3.2 MAIN CONFIGURATION

This tab is divided into multiple screens, all of which contain a set of global parameters regarding the general functionality of the device, and therefore not specifically related to a particular page of the user interface.

3.2.1 GENERAL

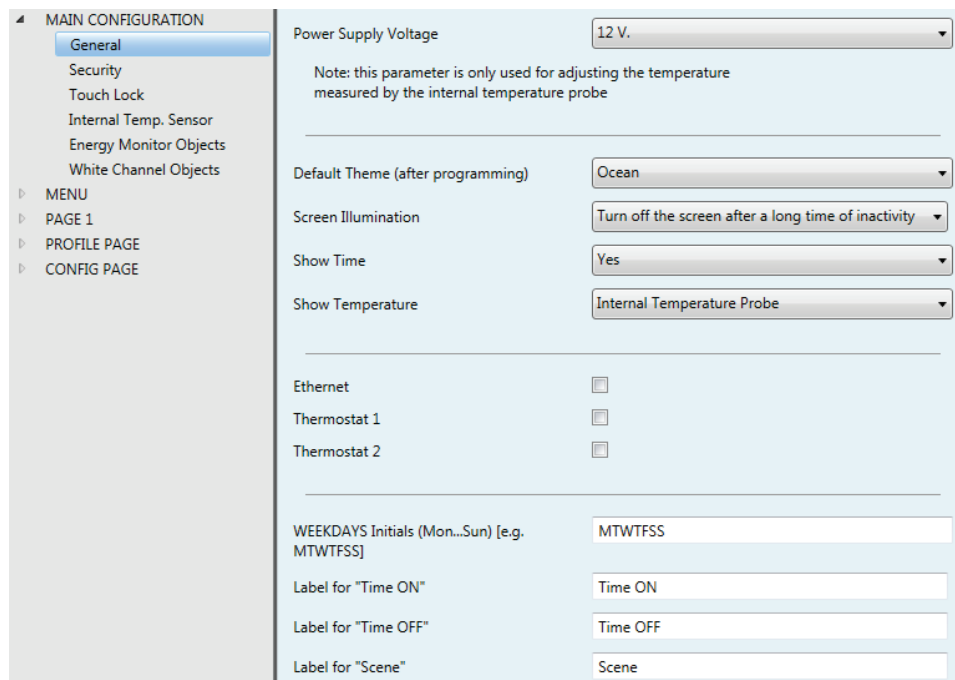


Figure 14 General (Main configuration)

The General screen contains the following parameters:

- **Power supply voltage:** lets the integrator specify the particular voltage of the external supply powering Z41. The available values are 12 (by default), 24 and 29 volts. Selecting one option or another will only cause an internal correction over the temperature value measured by the built-in probe.
- **Default theme (after programming):** permits selecting the user interface colour scheme (out of 8 different options) to be loaded after the ETS download. The final user will have the option to switch the scheme from the Profile page (see section 2.3), unless such option is disabled by parameter.
- **Show time:** sets whether the current time (according to the internal clock) should show in the upper left corner of every page. The default is "Yes".
- **Screen illumination:** lets defining whether the backlight should behave during inactivity according to section 2.5 (Turn off screen), or if on the contrary the screen should never turn off entirely (Attenuate the screen) or not even attenuate the light level (Max illumination always).

Note: *the default option is encouraged. Permanent illumination may affect adversely the device lifetime, and is only intended for special circumstances.*

- **Show temperature**: sets whether the current temperature should show or not in the upper right corner of every page, being necessary in such case to choose the source of the temperature value: “Internal temperature probe” or “External value”. If the latter is chosen, an object named **[General] External temperature** will be enabled, so that it can be grouped with any other object that sends temperature values.
- **Ethernet**: checkbox (unselected by default) that activates/deactivates the Ethernet parameter screen. See section 3.2.5.
- **Thermostat 1**: checkbox (unselected by default) that activates/deactivates the Thermostat 1 parameter screen. See section 3.7.
- **Thermostat 2**: checkbox (unselected by default) that activates/deactivates the Thermostat 2 parameter screen. See section 3.7.
- **Weekdays initials (Mon...Sun)**: field that permits customizing the label that represents each of the weekdays within the configuration pages of the weekly timers (see epigraph e) in section 3.4.2.5). A 7-character string (including letters or numbers) should be entered, so each of the characters will represent one weekday. The default value is “MTWTFSS”, according to the initial letters of the day names in English.
- **Label for “Time ON”**: field that permits customizing the label that will be shown within the configuration pages of the weekly timers (see epigraph e) in section 3.4.2.5), next to the checkbox that enables/disables the sending of the switch-on order. The default value is “Time ON”.
- **Label for “Time OFF”**: field that permits customizing the label that will be shown within the configuration pages of the weekly timers (see epigraph e) in section 3.4.2.5), next to the checkbox that enables/disables the sending of the switch-off order. The default value is “Time OFF”.
- **Label for “Scene”**: field that permits customizing the label that will be shown within the configuration pages of the weekly timers (see epigraph e) in section 3.4.2.5), next to the checkbox that enables/disables the sending of scenes. The default value is Scene”.

3.2.2 SECURITY

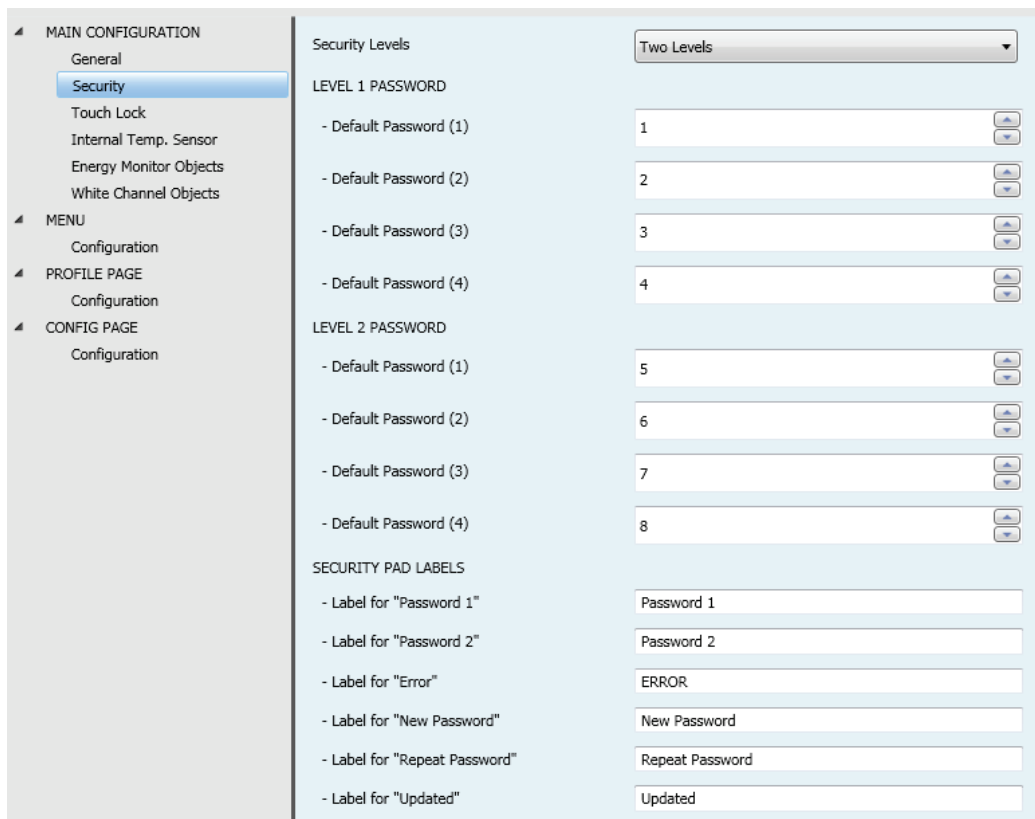


Figure 15 Security (Main configuration)

This screen permits selecting how many security levels (one or two) will be available for the configuration of the access to the control pages. In other words, it permits setting one or two different passwords, so the integrator can afterwards configure whether the access to each page will be protected by one password or another, or remain unprotected – every page can be independently configured. Figure 16 shows the “enter password” dialog shown to the user when trying to access a protected page.

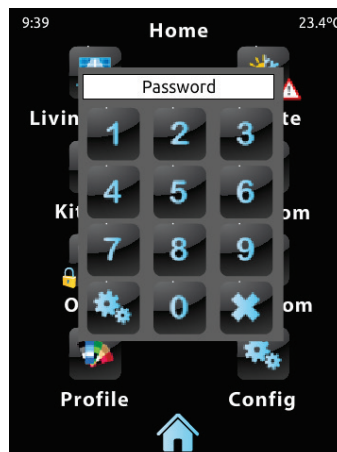


Figure 16 Password insertion dialog (for password protected pages)

In case of setting up **two levels**, the first one is assumed to be *enclosed* by the second one. This means that whenever the device asks the user to type in password #1 (to enter a certain page), password #2 will also be accepted. On the contrary, password #1 cannot be used instead of password #2. This behaviour permits, therefore, making password #2 available to users with further privileges while password #1 is assigned to users with fewer privileges.

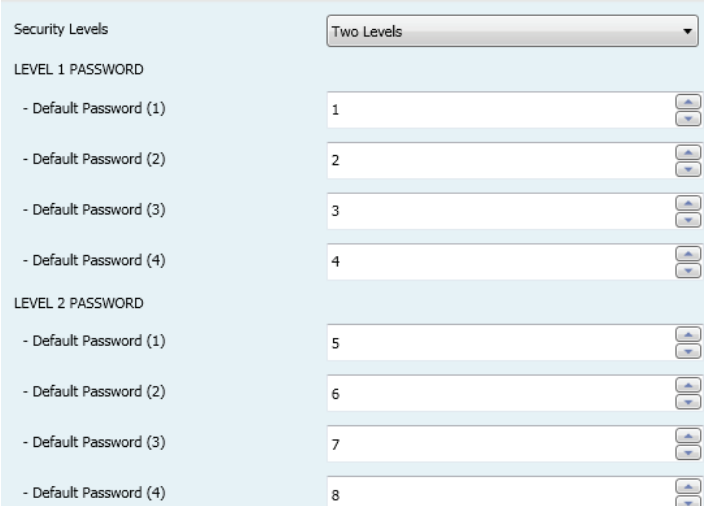
The parameters on the Security screen are:

- **Security levels:** dropdown list for selecting whether one (default option) or two security levels will be available. Depending on the selection, the parameter below (Password) may show once or twice.

Note: *with independence of the option selected here, later it will be necessary to establish the security level desired for each specific page.*

- **Password:** parameter made of four additional textboxes, each of which should contain one of the four consecutive digits (0-9) that will compose the password. The default password is “1234”.

In case of enabling two security levels, the Password parameter will show twice, being the first one referred to the password of Level 1, and the second one to the password of Level 2. The default password for Level 2 is “5678”.



The screenshot shows a configuration interface for security levels. At the top, there is a dropdown menu labeled 'Security Levels' with 'Two Levels' selected. Below this, the interface is divided into two sections: 'LEVEL 1 PASSWORD' and 'LEVEL 2 PASSWORD'. Each section contains four text input fields, each labeled '- Default Password (1)' through '(4)'. The first level's passwords are '1', '2', '3', and '4'. The second level's passwords are '5', '6', '7', and '8'. Each input field has a small up/down arrow button to its right.

Figure 17 Two security levels

Important: *the password insertion dialog features a specific option (lower left button) that lets the user change, in runtime, the passwords originally set by parameter. When the user accesses this option and prior to typing the new*

password, the user will be required to type the corresponding old password (level 1 or level 2). Note that although it will be possible to type password #2 even if the device asks for password #1, the new password typed afterwards will be anyway stored as the new password for level 1.

- **Security pad levels:** parameter consisting in six additional textboxes, intended for the customization of the messages that the device shows (or may show) when the user interacts with the password insertion dialog. Up to 15 characters are permitted per label.
 - **Label for “Password 1”:** message shown when the user is required to type in the password for level 1. By default, “Password 1”.
 - **Label for “Password 2”:** message shown when the user is required to type in the password for level 2. By default, “Password 2”.
 - **Label for “Error”:** message shown to the user when the typed password is not valid. By default, “ERROR”.
 - **Label for “New password”:** message shown to ask the user for a new password, during the password change process. By default, “New password”.
 - **Label for “Repeat password”:** message shown when the user is required to re-type the new password. By default, “Repeat password”.
 - **Label for “Updated”:** message shown to the user as a confirmation of the password change. By default, “Updated”.

Buttons that lead to a protected page from the Menu page (see section 2.1) will show a little lock icon overlaid on their lower left corner.

3.2.3 TOUCH LOCK

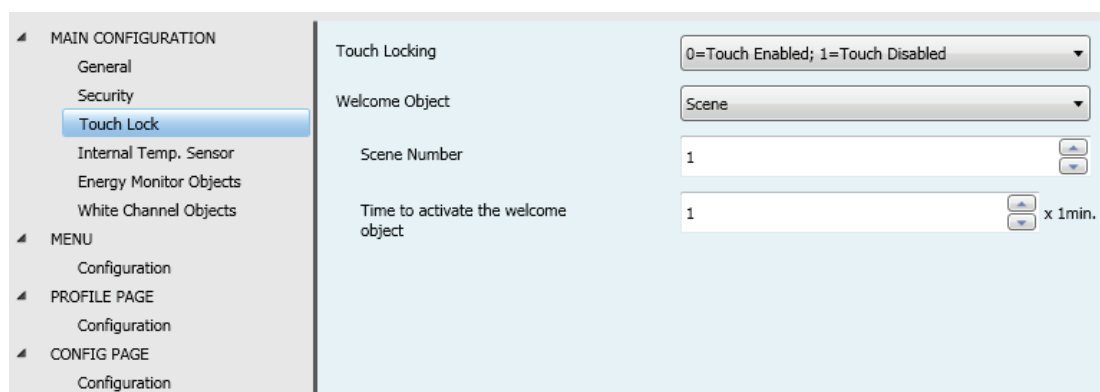


Figure 18 Touch lock (Main configuration)

This screen is provided for the configuration of the “welcome” and “touch lock” objects. By default two parameters are shown:

- **Touch locking:** dropdown list with the following options: “0=Touch enabled; 1=Touch disabled” (default value) and “0=Touch disabled; 1=Touch enabled”. Depending on the selection, sending one value or another (“0” or “1”) to object **[General] Touch lock** will lock/unlock the touch panel. The detailed behaviour of this object is described in section 3.1.
- **Welcome object:** dropdown list with the following options: “Disabled” (default value), “1-bit object” and “Scene”.
 - If “Disabled” is selected, this functionality will remain inactive. Therefore, nothing will happen when, after a certain standby period, the user resumes the interaction with the touch panel.
 - If “1-bit object” is selected, the **[General] Welcome object** binary object is enabled and two more parameters turn visible:
 - **Value:** value (“0” or “1”) that will be sent to the bus. By default it is “1”.
 - **Time to activate the welcome object:** minimum time of inactivity (in minutes) before a new press on the touch panel triggers the sending of the welcome object. By default, it is 1 minute.

| | |
|-------------------------------------|-----------------------------------|
| Touch Locking | 0=Touch Enabled; 1=Touch Disabled |
| Welcome Object | 1-bit object |
| Value | 1 |
| Time to activate the welcome object | 1 x 1min. |

➤ If “Scene” is selected, the **[General] Scenes: send** object, already enabled in the beginning, will work as the welcome object. In this case, the following two parameters come up:

- **Scene number:** number of the scene (1-64) that will be sent to the bus. The default value is “1”.
- **Time to activate the welcome object:** minimum time of inactivity (in minutes) before a new press on the touch panel triggers the sending of the parameterized scene. By default, it is 1 minute.

| | |
|-------------------------------------|-----------------------------------|
| Touch Locking | 0=Touch Enabled; 1=Touch Disabled |
| Welcome Object | Scene |
| Scene Number | 1 |
| Time to activate the welcome object | 1 x 1min. |

3.2.4 INTERNAL TEMPERATURE SENSOR

| | | |
|--|--------------------------------|------------------------|
| <ul style="list-style-type: none"> MAIN CONFIGURATION General Security Touch Lock Internal Temp. Sensor Energy Monitor Objects White Channel Objects MENU Configuration PROFILE PAGE Configuration CONFIG PAGE Configuration | Sensor Calibration | 0 x 0.1°C |
| | Sending Period | 0 x 10s. (0=Disabled) |
| | Send with a Temperature Change | 0 x 0.1°C (0=Disabled) |

Figure 19 Internal temperature sensor (Main configuration)

This screen permits configuring the internal temperature sensor of the device. Three parameters are provided for this:

- **Sensor calibration:** permits setting a certain correction over the values thrown by the internal temperature probe. This way, if the value of the

measurement is found to be slightly over (or below) the actual temperature due to external factors, it is possible to set an adjustment between -5.0°C and 5.0°C (0.0°C by default), in steps of 0.1°C . The value set here may be optionally changed, in runtime, from the Configuration page (see section 2.4).

- **Sending period:** cycle time (in tens of a second) for sending the value of the measured temperature to the bus. This sending is made through the **[General] Internal temperature** object (see section 3.1), enabled by default. The permitted values are 0 to 100 tens of a second. The value “0” (set by default) deactivates this sending to the bus.
- **Send with a temperature change:** permits performing a specific sending of the temperature value to the bus –through the **[General] Internal temperature** object– whenever an increase or decrease greater or equal than a certain value is detected between two consecutive measurements, no matter if a periodic sending has been enabled or not. Such value (unsigned) must be entered here, in terms of tenths of a degree. To avoid having this extra sending on temperature changes, simply leave the value “0” (default value).

3.2.5 ETHERNET

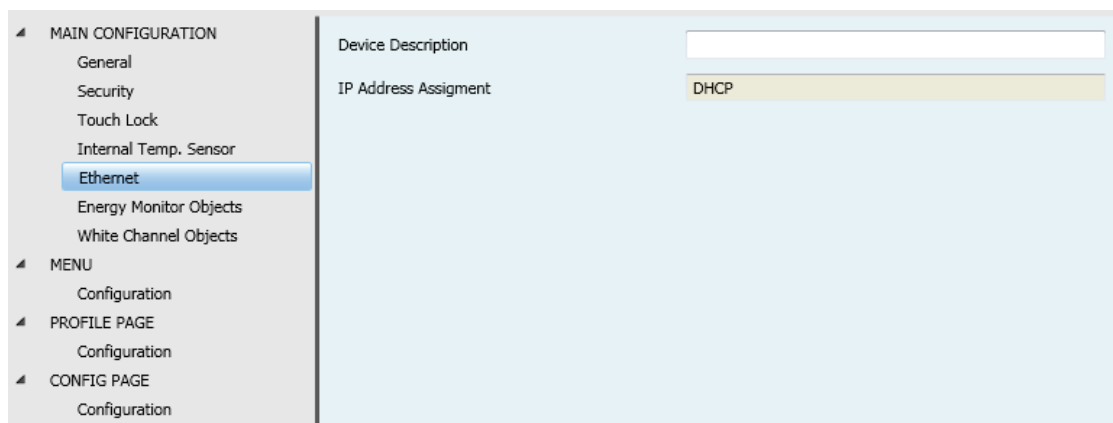


Figure 20 Ethernet (Main configuration)

The Ethernet screen, hidden by default, becomes visible once the corresponding checkbox has been marked under the General screen (see section 3.2.1).

The Ethernet interface in Z41 is intended, at the moment, for updating the firmware of the device (see section 1.3.1) across a local network. Alternatively, this process can be performed through the built-in USB interface.

From the Ethernet parameter screen it is possible to give a name or description to the device (**Device description**), so that identifying the device from the PC application for the firmware update via Ethernet turns easier, as this application may detect other Z41 devices connected to the local network.

Currently, Z41 only permits automatic IP address assignment (through the DHCP protocol). Therefore, the **IP Address Assignment** parameter implements no functionality at the moment.

3.2.6 ENERGY MONITOR OBJECTS

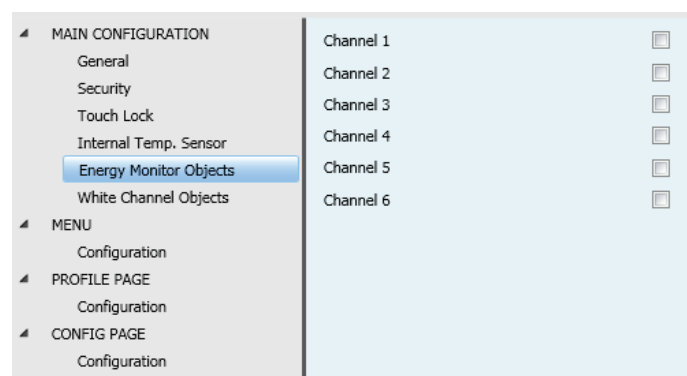


Figure 21 Energy monitor objects (Main configuration)

This screen contains six checkboxes, each of which permits enabling or disabling one communication “channel” that may be used for bus reception of energy consumption reports.

Every channel implements four communication objects:

- **[Energy x] Power:** 2-byte object to be linked to external objects reporting power values, such as the power measuring objects implemented by **KES**, the energy saver device from Zennio.
- **[Energy x] Energy:** 4-byte object to be linked to external objects reporting energy values, such as the energy measuring objects implemented by **KES**, the energy saver device from Zennio.
- **[Energy x] CO2:** 2-byte object to be linked to external objects reporting volumes of CO₂, such as those implemented by **KES** –the energy saver device from Zennio– to report the CO₂ emissions corresponding to the energy consumption.

- **[Energy x] Cost estimation:** 2-byte object to be linked to external objects that express, as floating point values, the cost of the energy consumption, in terms of the local currency. For example, the cost estimation objects implemented by **KES**, the energy saver device from Zennio.

So, when any of the six channels is enabled, the four communication objects corresponding to that channel will show up.

For further information about these objects and about power consumption monitoring in Z41, please refer to epigraph c) in section 3.4.2.5.

3.2.7 WHITE CHANNEL OBJECTS

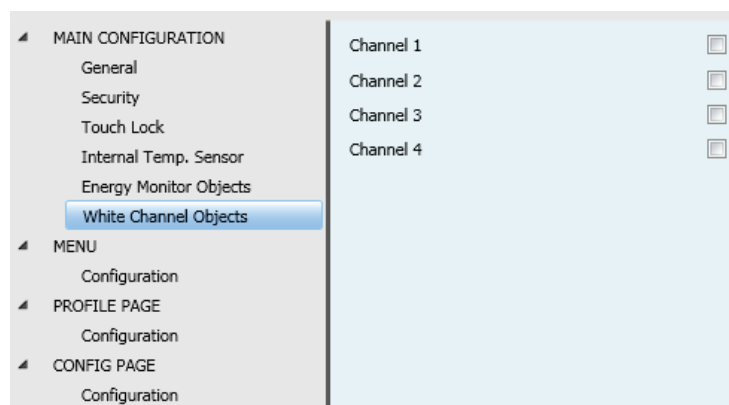


Figure 22 White channel objects (Main configuration)

This screen contains four checkboxes that allow the integrator to enable or disable up to four 1-byte communication objects independently. These objects, named **[RGBW] White channel x**, will be responsible for sending light dimming orders to the white channel of one or more per-channel RGBW lighting dimmers.

For further information about the usage of these objects and about the per-channel dimming controls implemented by Z41, please refer to epigraph b) in section 3.4.2.5.

3.3 MENU

The Menu tab contains only one screen, Configuration.

3.3.1 CONFIGURATION

Figure 23 Configuration (Menu)

The Configuration screen, under the Menu tab, permits configuring the Menu page itself, as well as a set of general options related to the enabled pages that can be accessed from the Menu page.

The available parameters are:

- **Title:** text field that defines the title that will be shown on the top of the Menu page.
- **Automatic page shaping:** dropdown list that sets whether the boxes in the Menu page should be automatically distributed (option “Yes”) depending on the amount of functional boxes, or display as a static 2x4 grid (option “No”). See section 2.6.
- **Default page:** dropdown list that sets the page (Menu, or any of the general-purpose pages) that will act as the default page. This page will be the one shown after one minute of inactivity, provided that such page has been enabled.

In addition, one checkbox is shown per general-purpose page (that is, pages 1 to 6), as well as two more checkboxes for the Profile and Configuration pages, making it possible to activate or deactivate every page from the user interface independently. A specific ETS tab will appear upon the activation of a page. The next sections of this manual describe these tabs in detail.

3.4 PAGE *n*

When any of the general-purpose pages is enabled from the Configuration screen under the Menu tab, a new tab named **Page *n*** will appear, where *n* is the number of the page. Under this tab, one screen (**Configuration**) will be initially displayed to let the integrator enable or disable any of the boxes in the page. Depending on that, more parameter screens will appear.

Note: figures in this section will show the parameterization of a certain page (for instance, page 1) or a certain box (for instance, box 1). The parameters for other pages or boxes should look similar.

3.4.1 CONFIGURATION

Figure 24 Configuration (Page *n*)

This screen contains the following parameters:

- **Title:** text field that defines the title of the page.
- **Automatic page shaping:** dropdown list that sets whether the boxes in the Menu page should be automatically distributed (option “Yes”) depending on the amount of functional boxes, or display as a static 2x4 grid (option “No”). See section 2.6.
- **Icon:** sets the icon that will represent the page in the Menu page.
Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

- **Protected:** dropdown list that permits setting whether the access to the page will be password protected or not. Depending on the security levels parameterized (one or two; see section 3.2.2) this list will contain the following options:

One level:

- **No:** no password protection for the page. All users can access it.
- **Yes:** the page will implement password protection. Users will be asked to type the password when trying to access it.

Two levels:

- **No:** no password protection for the page. All users can access it.
- **Level 1:** the page will implement security level 1. To access it, users will be required to enter password 1 or password 2.
- **Level 2:** the page will implement security level 2. To access it, users will be required to enter password 2.

Finally, eight checkboxes are provided to let the integrator turn functional or non-functional each of the eight boxes in the page. When functional, every box will have its own parameter screen (**Box *i***) under the corresponding **Page *n*** tab. The next section explains the parameterization process of these boxes.

3.4.2 BOX *i*

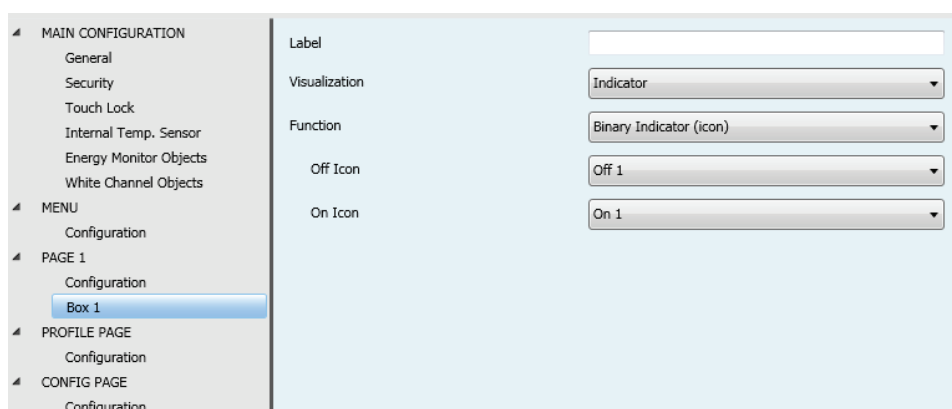


Figure 25 Box *i* (Page *n*)

This screen contains the following parameters:

- **Label:** title for the box.

- **Visualization:** box format, which may be: Indicator (default option; the box will simply work as a status indicator), 1-button control (the box will work as a control with one push button), 2-button control (the box will work not only as a status indicator, but also as a control with two push buttons), Climate control (the box will act as a climate indicator and control) or Other (the box will implement some other functionality).

Depending on the selected Visualization, the parameters will change. A detailed explanation for each can be found in the following sections.

3.4.2.1 INDICATOR



Figure 26 Indicator (box)

Boxes configured as indicators permit implementing a set of different functions. The desired function must be selected through the **Function** parameter. The available functions and the related parameters are:

a) Binary indicator (icon)

The box will behave as a binary state indicator. Each of the two states will be shown in the box through a different icon. When the box is assigned this function, the **[Pn][Bi] Binary indicator** communication object will turn visible, as well as two more parameters that permit selecting the icon to be displayed when the object acquires the value "0" (**Icon Off**) and the icon to be displayed when it acquires the value "1" (**Icon On**).

| | |
|---------------|-------------------------|
| Visualization | Indicator |
| Function | Binary Indicator (icon) |
| Off Icon | Off 1 |
| On Icon | On 1 |

Figure 27 Binary indicator (icon)

Therefore, when the device receives the values "0" or "1" through the aforementioned object, the box will show one icon or another.

Note: the list with all the icons available in Z41 can be found in the document "Z41 Icon list", downloadable from www.zennio.com.

b) Binary indicator (text)

The box will behave as a binary state indicator. Each of the two states will be shown in the box through a different label. When the box is assigned this function, the **[Pn][Bi] Binary indicator** object will turn visible, as well as two more parameters that permit typing the texts to be shown when the object acquires the value “0” (**Text Off**) and the text to be shown when it acquires the value “1” (**Text On**).

Therefore, when the device receives the values “0” or “1” through the aforementioned object, the box will show one text or another.

| | |
|---------------|-------------------------|
| Visualization | Indicator |
| Function | Binary Indicator (text) |
| Off Text | <input type="text"/> |
| On Text | <input type="text"/> |

Figure 28 Binary indicator (text)

c) Enumerated indicator (icon)

The box will behave analogously to the case of the Binary indicator (icon) however it will be possible to distinguish up to 6 states (configurable through the **# Enums** parameter) instead of only two. The states will be determined by the reception of any values between 0 and 255, not only by values 0 and 1.

When this function is assigned to the box, a 1-byte communication object, **[Pn][Bi] Enumerated indicator**, will turn visible as well as two more parameters (**Value** and **Icon**) for each of the states to be distinguished. This permits setting which icon will be shown in the box upon the reception of which value through the communication object.

| | |
|----------|-----------------------------|
| Function | Enumerated Indicator (icon) |
| # Enums | 2 |
| Value 1 | 0 |
| Icon 1 | Off 1 |
| Value 2 | 0 |
| Icon 2 | Off 1 |

Figure 29 Enumerated indicator (icon)

Example: a three-state enumerated indicator (“3” under “# Enum”) is parameterized as follows:

| | |
|---------------|------------------|
| Value 1 = "1" | Icon 1 = "One" |
| Value 2 = "3" | Icon 2 = "Two" |
| Value 3 = "5" | Icon 3 = "Three" |

When the **[Pn][Bi] Enumerated indicator** object receives the value "1", the box will show icon "One". When it receives the value "3", icon "Two" will be shown. And when it receives the value "5", icon "Three" will show. In case of receiving any other value, no icon will be shown.

d) Enumerated indicator (text)

The box will behave analogously to the case of the Binary indicator (text) however it will be possible to distinguish up to 6 states (configurable through the **# Enums** parameter) instead of only two. The states will be determined by the reception of any values between 0 and 255, not only by values 0 and 1. When this function is assigned to the box, one 1-byte communication object, **[Pn][Bi] Enumerated indicator**, will turn visible as well as two additional parameters (**Value** and **Text**) for each of the states to be distinguished. This permits setting which texts will be displayed in the box upon the reception of which values through the communication object.

Figure 30 Enumerated indicator (text)

e) Numerical indicators

Figure 31 1-byte indicator (unsigned integer)

The remaining functions available under Indicator are collected into Table 2. In all of these cases, the box will behave as a numerical state indicator that

permanently displays the value of a certain communication object, which is enabled when the function is assigned to the box.

| Function | Range | Related object |
|-------------------------|------------------------|--|
| 1-byte (unsigned int.) | 0– 255 | [Pn][Bi] 1-byte unsigned int indicator |
| 1-byte (signed int.) | -128 – 127 | [Pn][Bi] 1-byte signed int indicator |
| Percentage indicator | 0% – 100% | [Pn][Bi] Percentage indicator |
| 2-byte (unsigned int.) | 0 – 65535 | [Pn][Bi] 2-byte unsigned int indicator |
| 2-byte (signed int.) | -32768 – 32767 | [Pn][Bi] 2-byte signed int indicator |
| 2-byte (floating point) | -671088.64 – 670760.96 | [Pn][Bi] 2-byte float indicator |

Table 2 Numerical indicators

The table contains the allowed value ranges and the name of the object for each case.

In the particular case of the 2-byte floating point indicator, the integrator will be shown a text field (**Float unit**), empty by default, that permits specifying the measuring units (up to 6 characters) of the displayed value, for instance “°C”. Moreover, for this particular case, Z41 will dynamically adjust the number of the decimal figures shown depending on the order of magnitude of the current value: two for values greater than 0 and lower than 0.1; one for values between 0.1 and 100; and none for other cases (and analogously for negative values).

All these numerical indicators implement an additional parameter, **Save log?**. In case of enabling this option (“Yes”), any changes in the value of the indicator will be continuously saved to an internal log file, so the final user can be afterwards offered a visual representation that reflects the chronological evolution of such value. Boxes for which this option has been enabled will show a small icon overlaid on their upper left corner.

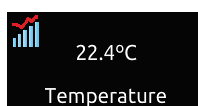


Figure 32 Indicator implementing the log function

Boxes showing this icon let the user press them, which will bring up a pop-up window similar to Figure 33.

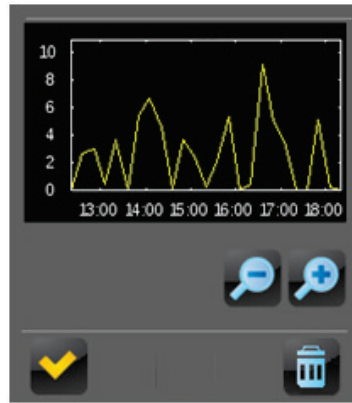


Figure 33 Graphical representation of the log file

The graph (where the vertical scale will dynamically adapt to the values being represented) shows the chronological evolution of the value, with higher or lower detail (hourly, daily, weekly, monthly; see Figure 34) depending on the zoom level set by the user by means of the on-screen magnifying glass control. In the case of the lower detail levels, the graph will show the average values per time unit (day or month) as well as the maximum and minimum values.

Finally, the bin icon (lower right corner of the screen) lets the user, by performing a long press, delete the contents of the log file, thus setting the values of the monitor to zero.

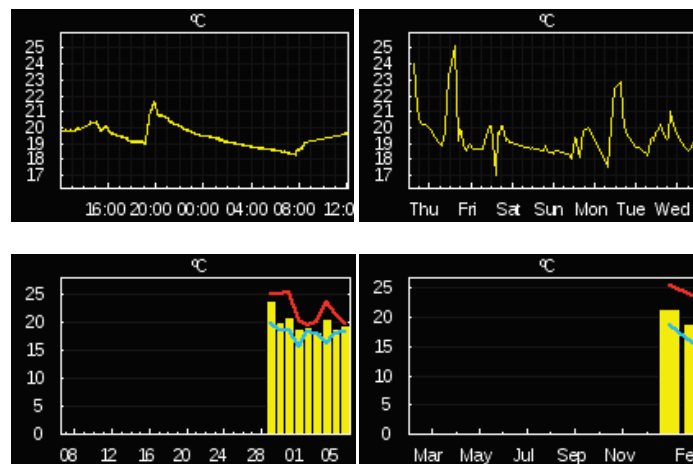


Figure 34 Different detail levels (log file)

3.4.2.2 1-BUTTON CONTROL

Boxes configured as 1-button controls show one centred button and a title.

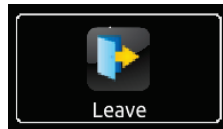


Figure 35 1-button control (box)

The desired function must be selected through the **Function** parameter. The available functions and the parameters related to each are:

a) Binary control

The central button of the box will react to user presses by sending a binary value to the bus through the **[Pn][Bi] Binary control** object, which turns visible as soon as this function is assigned to the box.

| | |
|------------------------|------------------|
| Visualization | 1-button Control |
| Function | Binary Control |
| Action | Toggle |
| Button (In the middle) | Off 1 |

Figure 36 Binary control

On the other hand, **Action** permits setting what value will be sent to the bus through the mentioned object, and on what events. The options are:

- **Toggle**: pressing the button will cause an alternating sending of the values “1” and “0”. Particularly, on the first press one “1” will be sent; on the second, one “0”; on the third, one “1”; and so on.
- **0**: one “0” will be sent whenever the button is pressed.
- **1**: one “1” will be sent whenever the button is pressed.
- **Short 1 – Long 0**: short presses will trigger the sending of the value “1”, while long presses will trigger the sending of the value “0”.
- **Short 0 – Long 1**: inverse to the previous case.
- **Hold 0 – Release 1**: one “0” will be sent when the button press starts, and one “1” as soon as the button is released.

- **Hold 1 – Release 0:** one “1” will be sent when the button press starts, and one “0” as soon as the button is released.

Finally, the dropdown list next to **Button (in the middle)** permits selecting the icon to be displayed inside the button.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

b) Numerical constant controls

If the box is assigned any of the remaining “constant” control options, the central button in the box will react to user presses by sending a certain numerical value, which is required to be specified under **Constant value**.

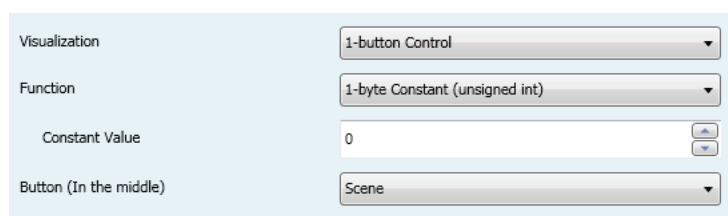


Figure 37 1-byte constant (unsigned int)

Table 3 collects, for every available Function, the permitted value range and the name of the object through which the values are sent to the bus.

| Function | Range | Object |
|---------------------------------|------------------------|--------------------------------------|
| 1-byte constant (unsigned int.) | 0 – 255 | [Pn][Bi] 1-byte unsigned int control |
| 1-byte constant (signed int.) | -128 – 127 | [Pn][Bi] 1-byte signed int control |
| Constant percentage value | 0% – 100% | [Pn][Bi] Percentage control |
| 2-byte constant (unsigned int.) | 0 – 65535 | [Pn][Bi] 2-byte unsigned int control |
| 2-byte constant (signed int.) | -32768 – 32767 | [Pn][Bi] 2-byte signed int control |
| 2-byte constant (float) | -671088.64 – 670760.96 | [Pn][Bi] 2-byte float control |

Table 3 Numerical constant controls

Finally, the dropdown list next to **Button (in the middle)** permits selecting an icon to be displayed inside the button.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

c) Scene control

In this case, the central button of the box will react to user presses by sending a scene value to the KNX bus through the **[General] Scenes: send** object, which is enabled by default. Parameter **Scene number** permits specifying the number of the scene (1 – 64) to be sent. Moreover, **Action** (which shows the options “Run” and “Run and save”) sets whether the device will only send scene execution orders (after a short press) or if it will be possible, in addition to sending execution orders upon short presses, to send scene save orders in the case of a long press.

| | |
|------------------------|------------------|
| Visualization | 1-button Control |
| Function | Scene Control |
| Scene Number | 1 |
| Action | Run |
| Button (In the middle) | Scene |

Figure 38 Scene control

Finally, the dropdown list next to **Button (in the middle)** permits selecting an icon to be displayed inside the button.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

3.4.2.3 2-BUTTON CONTROL

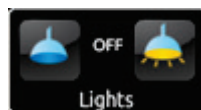


Figure 39 2-button control (box)

Boxes configured as 2-button controls consist in an indicator and two buttons that, when touched, trigger the sending of an action to the KNX bus through a certain object. This category also includes some specific controls such as **lighting** and **shutter** controls.

As a general rule, most of the 2-button controls permit configuring a pair of parameters, **Left button** and **Right button**, each containing a dropdown list for the selection of the desired icons to be displayed inside the buttons in the box.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

Note: when multiple presses are made consecutively on the buttons of a control that regulates (increases / decreases) the value of a certain status, only the final value selected by the user will be sent to the bus, so it is not overloaded by sending all the intermediate values.

On the other hand, the **Function** parameter contains a dropdown list for the selection of the particular two-button control type to be implemented by the box. The available options (and their related parameters) are:

a) Binary control (icon)

When the user touches the buttons, a binary value will be sent through the **[Pn][Bi] Binary control** object, while the status object, **[Pn][Bi] Binary indicator**, will determine the icon shown in the box.

| | |
|---------------|-----------------------|
| Visualization | 2-button Control |
| Function | Binary Control (icon) |
| Action | Left 0, Right 1 |
| Off Icon | Off 1 |
| On Icon | On 1 |
| Left Button | Off 1 |
| Right Button | On 1 |

Figure 40 Binary control (icon)

The available parameters are:

- **Action:** sets the value to be sent on button touches. This can be “Left 0, Right 1” (by default) or “Left 1, Right 0”.
- **Off icon** and **On icon:** dropdown lists for the selection of the icons that will represent the two states (0 / 1) of the **[Pn][Bi] Binary indicator** object, which gets automatically updated whenever an order is sent through the control object. It may also receive values from the bus.

b) Binary control (text)

When the user touches the buttons, a binary value will be sent through the **[Pn][Bi] Binary control** object, while the status object, **[Pn][Bi] Binary indicator**, will determine the text that will be shown in the box.

| | |
|---------------|-----------------------|
| Visualization | 2-button Control |
| Function | Binary Control (text) |
| Action | Left 0, Right 1 |
| Off Text | |
| On Text | |
| Left Button | Off 1 |
| Right Button | On 1 |

Figure 41 Binary control (text)

The available parameters are:

- **Action:** sets the value to be sent on button touches. This can be “Left 0, Right 1” (default) or “Left 1, Right 0”.
- **Text Off** and **Text On:** indicative texts that will be shown in the box depending on the state (0 / 1) of the **[Pn][Bi] Binary indicator** object, which gets automatically updated whenever an order is sent trough the control object. It may also receive values from the bus.

c) Enumerated control (icon)

The box will behave analogously to the case of the Binary control (icon) however it will be possible to distinguish up to 6 states (configurable through the **# Enums** parameter) instead of only two. The states will be determined by the reception of any values between 0 and 255, as the control (**[Pn][Bi] Enumerated control**) and the status (**[Pn][Bi] Enumerated indicator**) objects are not binary but 1-byte objects.

The available parameters are:

- **# Enums:** number of states (up to 6) that will be distinguished. For every distinguished state, the parameters **Value** and **Icon** will become available, in order to relate every significant value of the communication object to a certain icon that will represent it.

| | |
|---------------|---------------------------|
| Visualization | 2-button Control |
| Function | Enumerated Control (icon) |
| # Enums | 1 |
| Value 1 | 0 |
| Icon 1 | Off 1 |
| Left Button | Arrow Left |
| Right Button | Arrow Right |

Figure 42 Enumerated control (icon)

- **Value j** : numerical value (0 – 255) that will be sent through the control object when the user, after pressing the buttons, sets the control box to state j .
- **Icon j** : indicative icon that will be displayed in the box when the status object (which gets automatically updated whenever a control order is sent, although it may also receive values from the bus) acquires the numerical value defined in the above parameter.

d) Enumerated control (text)

The box will behave analogously to the case of the Binary control (text) however it will be possible to distinguish up to 6 states (configurable through the **# Enums** parameter) instead of only two. The states will be determined by the reception of any values between 0 and 255, as the control (**[Pn][Bi] Enumerated control**) and the status (**[Pn][Bi] Enumerated indicator**) objects are not binary but 1-byte objects.

| | |
|---------------|---------------------------|
| Visualization | 2-button Control |
| Function | Enumerated Control (text) |
| # Enums | 1 |
| Value 1 | 0 |
| Text 1 | |
| Left Button | Arrow Left |
| Right Button | Arrow Right |

Figure 43 Enumerated control (text)

The available parameters are:

- **# Enums:** number of states (up to 6) that will be distinguished. For every distinguished state, the parameters **Value** and **Icon** will become available, in order to relate every significant value of the communication object to a certain icon that will represent it.
- **Value j :** numerical value (0 – 255) that will be sent through the control object when the user, after pressing the buttons, sets the control box to state j .
- **Text j :** indicative text that will be displayed in the box when the status object (which gets automatically updated whenever a control order is sent, although it may also receive values from the bus) acquires the numerical value defined in the above parameter.

e) Numerical controls

In the case of assigning the box any of the six numerical functions, user touches over the buttons will trigger the sending of a certain numerical value to the bus (this value will be progressively increased or decreased every time the user touches one button or the other) through the control object, while the box itself will permanently reflect the current numerical value of the status object (which gets automatically updated after a control order is sent to the bus, although it may also receive values from the bus).

Table 4 shows, for each of the six functions, the permitted value range and the name of the control and status objects.

| | |
|-----------------|-------------------------------|
| Visualization | 2-button Control |
| Function | 1-byte (unsigned int) |
| Action | Left Decrease, Right Increase |
| Initial Value | 0 |
| Minimum Value | 0 |
| Maximum Value | 255 |
| Increment Short | 1 |
| Increment Long | 10 |
| Left Button | Minus |
| Right Button | Plus |

Figure 44 1-byte (unsigned int.) 2-button control

| Range | Control object | Status object |
|-------------------------------|--------------------------------------|--|
| 1-byte (unsigned int.) | | |
| 0 – 255 | [Pn][Bi] 1-byte unsigned int control | [Pn][Bi] 1-byte unsigned int indicator |
| 1-byte (signed int.) | | |
| -128 – 127 | [Pn][Bi] 1-byte signed int control | [Pn][Bi] 1-byte signed int indicator |
| Percentage control | | |
| 0% – 100% | [Pn][Bi] Percentage control | [Pn][Bi] Percentage indicator |
| 2-byte (unsigned int.) | | |
| 0 – 65535 | [Pn][Bi] 2-byte unsigned int control | [Pn][Bi] 2-byte unsigned int indicator |
| 2-byte (signed int.) | | |
| -32768 – 32767 | [Pn][Bi] 2-byte signed int control | [Pn][Bi] 2-byte signed int indicator |
| 2-byte (float) | | |
| -671088.64 – 670760.96 | [Pn][Bi] 2-byte float control | [Pn][Bi] 2-byte float indicator |

Table 4 Numerical controls

For all the six functions, the available parameters are:

- **Action:** sets which of the two buttons will be increasing the current numerical value and which will be decreasing it on user presses. The permitted options are “Left Decrease, Right Increase” (default) and “Left increase, Right Decrease”.
- **Initial value:** sets which value from the permitted range will be assumed by the control prior to any button presses. It is always 0 by default.
- **Minimum value:** sets which value from the available range will the minimum value permitted by the control after a number of presses on the decrease button.
- **Maximum value:** sets which value from the available range will the maximum value permitted by the control after a number of presses on the increase button.
- **Short increment:** sets the increase or decrease to be applied to the current value on every short press over the increase or decrease buttons, respectively (i.e., the smaller the increment is, the more presses will be

required). It is 1 by default, except for the case of the 2-byte floating point control, where it is 0.5.

- **Long increment:** sets the increase or decrease to be applied to the current value on every long press over the increase or decrease buttons, respectively (i.e., the smaller the increment is, the more presses will be required). It is 10 by default, except for the case of the 2-byte floating point control, where it is 1.

f) Two-button scene control

Analogously to the 1-button scene control, 2-button scene controls permit sending the KNX bus a scene value through the **[General] Scenes: send** object, however in this case there will be two buttons in the box, each of which can be configured independently.

| | |
|---------------|--------------------------|
| Label | <input type="text"/> |
| Visualization | 2-button Control |
| Function | Two-button Scene Control |
| Left Scene | 1 |
| Action | Run and Save |
| Button Icon | Scene |
| Right Scene | 1 |
| Action | Run and Save |
| Button Icon | Scene |

Figure 45 2-button scene control

Through the **Scene** parameter it is possible to assign each of the two buttons (left / right) the number of the scene to be sent (1 – 64). In addition, **Action** (which permits the options “Run” and “Run and save”) lets the integrator set whether the device will only send execution orders for the specified scene (after a short press), or it will possible, apart from sending execution orders on short presses, to also send save orders for such scene, in the case of performing a long press on the button.

Last, the dropdown list under **Button icon** sets the icon to be displayed by the buttons.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

g) Shutter control

Shutter controls permit sending move up, move down, stop or step shutter orders to the KNX bus by pressing the buttons in the box. In addition, the box will contain an indicator that will permanently show, as a percentage, the value of the status object.

| | |
|---------------|---------------------|
| Visualization | 2-button Control |
| Function | Shutter Control |
| Action | Left Down, Right Up |
| Left Button | Shutter Down 1 |
| Right Button | Shutter Up 1 |

Figure 46 Shutter control

The only specific parameter for this function is:

- **Action:** sets which of the two buttons will send the move up orders and which the move down orders. Options are “Left Down, Right Up” (default) and “Left Up, Right Down”.

Setting up a box as a shutter control enables three communication objects, **[Pn][Bi] Shutter position** (1 byte), **[Pn][Bi] Move shutter** and **[Pn][Bi] Stop shutter** (both, binary objects). The former acts as the status object, which determines the value shown in the box; it will be necessary to link it to the analogous object from the shutter actuator (as it does not get automatically updated on user presses) so it receives the value 100% when the shutter is completely down, and the value 0% when it is totally up. On their side, control orders will be sent through one of the two remaining objects:

- Orders to move the shutter up/down (completely) are sent through **[Pn][Bi] Move Shutter** after a long press on the corresponding button. Value “0” represents the move up order, while “1” represents the move down order.
- Orders to perform a step movement (value “0” for steps up and “1” for steps down) are sent through **[Pn][Bi] Stop shutter** after a short press on

the corresponding button. If the shutter was already in movement, these orders are interpreted by the actuator as a stop order.

h) Lighting control

The lighting control function permits making use of the two buttons in the box to send orders to a light dimmer, either through a binary object or through a 4-bit object. Moreover, the box will permanently display the current value of the dimming status object, which needs to be linked to the analogous object from the dimmer (as it does not get automatically updated on button presses).

| | |
|---------------|--------------------|
| Visualization | 2-button Control |
| Function | Light Dimming |
| Action | Left Off, Right On |
| Dimming Step | 100% |
| Left Button | Light Off 1 |
| Right Button | Light On 1 |

Figure 47 Lighting control

The available parameters are:

- **Action:** sets which of the two buttons will send the “turn on” orders and which the “turn off” orders. Options are “Left Off, Right On” (default) and “Left On, Right Off”.
- **Dimming step:** sets the increase or decrease in the light level (in terms of percentage) the dimmer must perform with every step order. How step dimming works is detailed below.

Setting up a box as a dimming control brings up the objects **[Pn][Bi] Light indicator** (1 byte), **[Pn][Bi] Light On/Off** (1 bit) and **[Pn][Bi] Light dimming** (4 bits). The former acts as the dimming status object, which is required to be linked to the analogous object from the dimmer and whose value, in percentage, will be permanently reflected in the box. On their side, control orders will be sent through one of the two remaining objects:

- After a short press on the “turn on” button, the value “1” will be sent through the **[Pn][Bi] Light On/Off** binary object, while a short press on the “turn off” button will trigger the sending of the value “0”.

- When a long press on the “turn on” button takes place, the object **[Pn][Bi] Light dimming** will send the light increment order corresponding to the parameterized dimming step, while a stop order (value “8”) will be sent through the same object as soon as the user releases the button.
- When a long press on the “turn off” button takes place, the object **[Pn][Bi] Light dimming** will send the light decrement order corresponding to the parameterized dimming step, while a stop order (value “0”) will be sent through the same object as soon as the user releases the button.

Table 5 shows the number of steps (that is, of long presses) required for a complete light regulation (from totally off to totally on, or vice versa) depending on the value parameterized as the dimming step.

| Dimming step | Button presses required for complete 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 5 Dimming steps

Note: *most light dimmers implement light step dimming progressively (i.e., sending a dimmer a step order of 25% typically does not imply that the light level is suddenly incremented/decremented by 25%, but a progressive increment or decrement of the light level by 25% which is in fact interrupted if a stop order arrives (such order is sent by Z41 when the user releases the button). Due to this behaviour, it is advised to parameterize dimming steps of 100%, so that the user can perform a complete dimming (from totally off to totally on, or vice versa) or a partial dimming by simply holding the button and then releasing it as soon as he gets the desired light level, therefore with no need of performing successive long presses for regulations greater than the parameterized step.*

3.4.2.4 CLIMATE CONTROL

This category covers a set of functions related to the climate control. The available options for **Function** and for the dependent parameters are as follows:

a) Setpoint control

This function permits controlling the temperature setpoint of an external thermostat by means of a two-button box and of the parameters enabled to that effect: **[Pn][Bi] Temperature control** for the control and **[Pn][Bi] Temperature indicator** for the status. In addition, the box itself will permanently reflect the value (in °C) of the status object, whose value gets automatically updated after sending control orders (that is, after pressing the buttons), being even possible to receive values from the bus, for example, from the corresponding setpoint status object from the external thermostat.

| | |
|---------------|-------------------------------|
| Visualization | Climate Control |
| Function | Setpoint Control |
| Action | Left Decrease, Right Increase |
| Setpoint Type | Absolute |
| Minimum Value | 10 x 1°C |
| Maximum Value | 30 x 1°C |
| Left Button | Temp Decrease |
| Right Button | Temp Increase |

Figure 48 Setpoint control

Therefore, after every press on the temperature increment button, the bus will be sent (through the control object) a certain value, progressively increased on every button press until the parameterizable maximum setpoint value has been reached. Analogously, after every press on the temperature decrement button, the bus will be sent a progressively decreased value until the parameterizable minimum setpoint value has been reached.

The available parameters are:

- **Action:** sets which of the two buttons will permit increasing the setpoint value and which one will permit decreasing it. Options are “Left decrease, Right increase” (default) and “Left increase, Right decrease”.

- **Setpoint type:** it may be absolute (default), relative (1-bit object) or relative (float object).

If the first is selected, the control object will be a 2-byte floating point object and its value will correspond to the actual setpoint value to be sent to the thermostat, being progressively higher or lower as the user touches the control buttons. On the other hand, the permitted setpoint temperatures will be delimited by a **maximum value** and a **minimum value**, defined by the so-named parameters.

Figure 49 Absolute setpoint control

If the second option is selected (1-bit object setpoint control), the control object will be binary, and it will throw a “0” to the bus whenever the user asks for a setpoint decrement, and a “1” when the user asks for an increment. How much it is incremented or decreased after every order is defined through the **offset** parameter (which accepts values between 1 and 255 tenths of a degree), thus permitting a consistent self-update of the label in the box after every user press, although the status object will also admit the reception of feedback from the bus, i.e., the updated absolute setpoint value sent from the external thermostat.

Figure 50 Relative (1 bit) setpoint control

Finally, when “Relative (float object)” is selected, the control will send 2-byte floating point values. However, these values will correspond to the particular increment or decrement (or offset) to be applied each time. This offset refers to a base temperature and will always be a multiple of 0.5°C. As the user touches the increase/decrease buttons in the box, this offset will vary, always within a certain range defined through the **Minimum offset** and **Maximum offset** parameters.

| | |
|----------------|-------------------------|
| Setpoint Type | Relative (float object) |
| Minimum Offset | -10 x 1°C |
| Maximum Offset | 10 x 1°C |

Figure 51 Relative (float object) setpoint control

Example: a floating point relative setpoint control is parameterized with a minimum offset of -10 and a maximum offset of 10, and then the objects are linked to those from the external thermostat, which can be supposed to have a base setpoint temperature of 25°C. After the parameter download, the first touch on the increase button will trigger the sending of the value “0.5”, so that the thermostat sets the setpoint to 25.5°C. This value will be then sent through its status object, and afterwards shown by Z41 in the corresponding box. A second touch will trigger the sending of a “1”, after which the setpoint will become 26°C. On the third touch, the sent value will be “1.5”, so the setpoint will be set to 26.5°C. And so on until the offset sent becomes 10 (setpoint of 35°C). Any further presses on the increase button will send the value “10” again, so the setpoint remains at 35°C, while touching the decrease button for the first time will trigger the sending of the value “9.5”, setting the temperature back to 24.5°C. On the second button press, the value “9” will be sent, and the setpoint will become 24°C. And so on until an offset of “-10” (setpoint of 15°C) is reached. From that moment, any further presses on the decrease button will re-send the value “-10” and the setpoint will remain at 15°C. Hence, this type of control permits regulating the setpoint temperature 10°C (for example) above or below a certain base setpoint temperature.

Important: the control object for the floating point relative setpoint control has its Write flag enabled with the intention of permitting the reception of feedback from the thermostat, which may reset or modify the status of the offset for any reason at any time. This feedback allows that Z41 updates its own value to equal that from the external thermostat. Thinking of the above example, if Z41 receives the value “0” after having the user performed three presses on the increase button (therefore, after having

been sent the value “1.5” to the bus), it will happen that on the forth press the value sent will be “0.5” and not “2”.

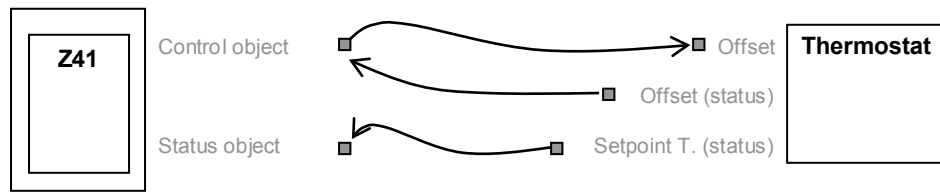


Figure 52 Relative (float) setpoint control. Object linking.

Note: the parameterized setpoint type becomes transparent to the final user, as the aspect of the on-screen box remains the same in all cases. In addition, the setpoint status object is always a 2-byte floating point object and will represent, in any case, the absolute value of the current setpoint temperature.

- **Left button** and **Right button:** permit selecting the desired icon to be displayed inside each button in the box.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

b) Mode

This function turns the box into a climate mode control.

| | |
|---------------|-----------------|
| Visualization | Climate Control |
| Function | Mode Control |
| Mode Type | Heat/Cool |
| Left Button | Mode Cool |
| Right Button | Mode Heat |

Figure 53 Heat/Cool mode type

Two mode control types are possible: the **heat / cool** control type and the **extended** (HVAC) control type. Selecting the former or the latter is possible through the **Mode type** parameter.

➤ **Heat / cool**

Selecting this type of mode control turns the box into a 2-button control one of which will activate the Heat mode, while the other one will activate the Cool mode. A centred indicator will reflect, as an icon, the currently active mode.



Figure 54 Heat/cool mode (box)

Two communication objects are also enabled: the **[Pn][Bi] Mode control** control object and the **[Pn][Bi] Mode indicator** status object. When the user activates the Cool mode, the device will send the value “0” through the control object, while on the activation of the Heat mode the value “1” will be sent. Moreover, the box will automatically commute between one icon and the other every time the user executes a mode change, and also on the reception of a new value (from the bus) through the status object.

➤ **Extended**

Selecting this type of mode control turns the box into a 2-button control, which permit sequentially commuting among the different HVAC climate modes. A centred indicator will reflect, as an icon, the currently active mode.



Figure 55 Extended mode (box)

Up to five modes (Heat, Cool, Auto, Fan and Dry) are available, each of which can be enabled/disabled in ETS by means of the proper checkbox, which permits setting which of all the five extended modes will be included into the sequential scrolling implemented by the buttons.

Figure 56 Extended mode

When this type of mode control is assigned to the box, two 1-byte communication objects are enabled: the **[Pn][Bi] Mode control** control object, and the **[Pn][Bi] Mode indicator** status object. Depending on the mode selected by the user, the control object will be sent to the bus a certain value (see Table 6), after which the box will display the icon that corresponds to the new mode. Additionally, if the status object receives from the bus a value that represents any of the modes, the box will adopt the corresponding icon, while if an unrecognised value is received, no icon will be shown.

| Mode | Value sent |
|------|------------|
| Heat | 1 (0x01) |
| Cool | 3 (0x03) |
| Auto | 0 (0x00) |
| Fan | 9 (0x09) |
| Dry | 14 (0x0E) |

Table 6 HVAC mode vs. object value

Finally, the **Left button** and **Right button** parameters permit selecting the icon that will be displayed inside the buttons of the box.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

c) Fan

This function implements a 2-button (increase / decrease) fan control, as well as an icon indicator.

| | |
|---------------|-------------------------------|
| Visualization | Climate Control |
| Function | Fan Control |
| Action | Left Decrease, Right Increase |
| Fan Type | Min/Max |
| Left Button | Minus |
| Right Button | Plus |

Tabla 7 Fan control

When this function is assigned to the box, a binary control object (**[Pn][Bi] Fan control**) and a 1-byte status object (**[Pn][Bi] Fan indicator**) are enabled. The former will be sending the value “1” to the bus when the user touches the increase button, while the value “0” will be sent when the decrease button is touched. On the other hand, the status object (which needs to be linked to the status object of the fan actuator) will express, as a percentage, the value of the current fan level, which will be represented with a variable icon on the box.

The available parameters are:

- **Action:** permits setting the increase or decrease actions to one button or another. Options are: “Left: decrease, Right increase” (default) and “Left increase, Right decrease”.
- **Fan type:** defines the number of states to be distinguished depending on the value of the status object, so more or less different icons will be used to represent the fan level in the box.
 - Min / Max: the first state (**minimum**) will correspond to any value within the range 0% to 49%, both included; the second state (**maximum**) will correspond to all values greater than 49%.
 - Min / Med / Max: levels between 0% and 33%, both included, correspond to the **minimum** state; values between 34% and 66%, to the **medium** state; and values over 67% to the **maximum** state.
 - Auto / Min / Med / Max: a level of 0% will be interpreted as the **Auto** state; levels between 1% and 33%, both included, correspond to the **minimum state**; levels between 34% and 66% will be considered as the **medium** state; and a value of 67% or more will correspond to the **maximum** state.

- Off / Min / Med / Max: 0% will be associated to the **off** fan state; between 1% and 33%, both included, to the **minimum** state; values between 34% and 66% will be interpreted as the **medium** state; and 67% and over will correspond to the **maximum** state.
- **Left button** and **Right button**: permit selecting the icon that will be displayed inside the buttons of the box.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

d) Special modes



Figure 57 Special modes (box)

Boxes configured as special mode controls include two buttons that let the user sequentially commute between the different special climate modes, as well as an icon indicator that will adopt an aspect or another depending on the currently active special mode.

| | |
|-----------------------|----------------------|
| Visualization | Climate Control |
| Function | Special Mode Control |
| Allow Protection Mode | No |
| Left Button | Arrow Left |
| Right Button | Arrow Right |

Tabla 8 Special modes

When this function is assigned to the box, two 1-byte objects (**[Pn][Bi] Special mode control** as the control object, and **[Pn][Bi] Special mode indicator** as the status object) will be enabled. The former will be sending the bus the value that corresponds to the mode that the user selects by touching the buttons (see Table 9), while the status object (which can receive values from the bus although it gets self-updated as well on button presses) will determine the indicative icon represented in the box at any time. If this object receives an unrecognised value from the bus, no icon will be displayed in the indicator.





| Special mode | Icon | Object value |
|--------------|---|--------------|
| Comfort |  | 1 (0x001) |
| Standby |  | 2 (0x002) |
| Economy |  | 3 (0x003) |
| Protection |  | 4 (0x004) |

Table 9 Special modes vs. icon vs. object value

The remaining parameters for this function are:

- **Allow protection mode:** this parameter permits including (“Yes”, default option) or excluding (“No”) the “building protection” mode from the special modes listed in the sequential scrolling performed by the buttons.
- **Left button** and **Right button:** permit selecting the icon that will be displayed inside the buttons in the box.

Note: the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.

3.4.2.5 OTHER

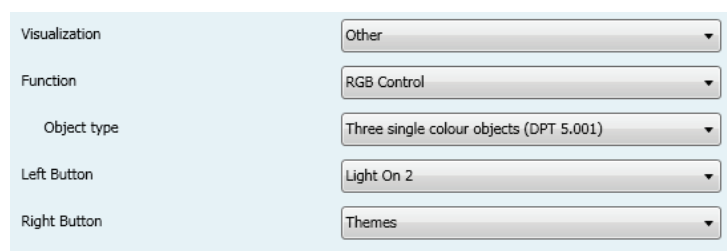


Figure 58 Other

This category involves the following functions, which can be selected through the **Function** parameter.

a) **RGB**

This function is intended for sending orders to three-colour LED light regulators, such as Lumento X3 RGB from Zennio.

| | |
|--------------|---|
| Function | RGB Control |
| Object type | Three single colour objects (DPT 5.001) |
| Left Button | Light On 2 |
| Right Button | Themes |

Figure 59 RGB

When the function is assigned to the box, the following parameters come up:

- **Object type:** permits selecting what type of object will be used for controlling the light level of the RGB channels. It may be “Three single colour objects (DPT 5.001)” (by default) or “One RGB object (DPT 232.600)”. If the former is selected, three 1-byte objects will be enabled (with their Write and Read flags activated, so they can work as control and status objects) named:

- **[Pn][Bi] Red channel**
- **[Pn][Bi] Green channel**
- **[Pn][Bi] Blue channel**

Control orders will consist in sending the light level (as a percentage) of the different channels through the corresponding objects.

If the second option is selected, only one 3-byte object will be enabled:

- **[Pn][Bi] RGB Colour**

In this case, the light levels of the three channels are sent (and received) concatenated into the above 3-byte object.

- **Left button** and **Right button:** permit selecting the icon that will be displayed inside the buttons in the box.

Note: *the list with all the icons available in Z41 can be found in the document “Z41 Icon list”, downloadable from www.zennio.com.*

Regarding the RGB control boxes themselves, a central label will permanently show, as a percentage, the current light level (which is determined by the channel with the highest current light level). This **indicator** gets updated automatically as the user interacts with the box, but is also conditioned by the values received from the bus through the already described objects.



Figure 60 RGB (casilla)

Moreover, two buttons can be found in the box:

- The button on the left, which is an **On/Off** switch, permits commuting between a total absence of light in the three channel, and their respective states previous to the switch-off.
- The button on the right, when pressed, launches a **colour palette** that permits the selection of a light level for every channel, either by touching on any of the preset colours, or by separately selecting the level of each channel (R, G, B).

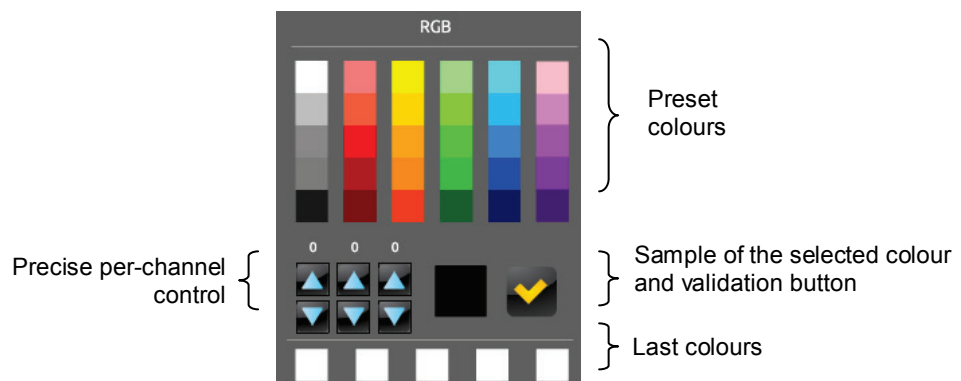


Figure 61 RGB colour palette

A set of recently used colours is shown at the bottom of the palette. These colours refer to those recently used within that particular box.

b) **RGBW**

Although totally analogous to the function just described, the RGBW control additionally permits controlling one extra channel specifically for white light, intended for dimmers that implement such function, such as Lumento X4 LED from Zennio.

| | |
|---|---|
| Visualization | Other |
| Function | RGBW Control |
| Object type | Three single colour objects (DPT 5.001) |
| RGBW Channel | Channel 1 |
| Enable the selected channel in "Main Config./White Channel Objects" tab | |
| Left Button | Light On 2 |
| Right Button | Themes |

Figure 62 RGBW

Therefore, the functionality is similar to that of the RGB control (including the possibility of controlling the RGB channels both through binary objects or through a sole 3-byte object), however one additional parameter (**RGBW Channel**) is provided to let the integrator select which of the four white channels available will be regulated by the box. The channel selected here must have been specifically enabled from the **White channel objects** screen (see section 3.2.7) under the Main configuration tab, which will make the **[RGBW] White channel x** available for sending dimming orders to the white channel.

Important: *although it is possible to assign the "RGBW control" function to any box from any page, it is only permitted to control as many as four different white channels, as it is not possible to have a different **[RGBW] White channel x** for each of the boxes. This restriction does not apply to the R, G and B channels from the RGB or RGBW controls: each box will make use of its own communication objects for regulating such channels.*

Finally, RGBW control boxes look similarly to RGB control boxes:



Figure 63 RGBW (box)

Nevertheless, when the colour palette of an RGBW box is brought up, a forth increase/decrease regulator is shown together with the three regulators shown in the case of the RGB control boxes:



Figure 64 RGBW colour palette

c) Energy monitor (KES)

| | |
|--|--------------------------|
| Visualization | Other |
| Function | Energy Monitor (KES) |
| Energy Monitor Channel | Channel 1 |
| Enable the selected channel in "Main Config./Energy Monitor Objects" tab | |
| Energy Consumption | <input type="checkbox"/> |
| CO2 | <input type="checkbox"/> |
| Cost | <input type="checkbox"/> |

Figure 65 Energy monitor (KES)

This function permits making use of one of the boxes of Z41 as a monitor of information related to energy consumption –which may be reported by devices such as KES from Zennio– and in particular, as a monitor of the evolution of the instant power.

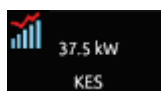


Figure 66 Energy monitor (KES). Box.

When the user touches any of these boxes, a pop-up window similar to Figure 67 will be shown.

Prior to detailing the elements shown in the pop-up window, the ETS parameters for this type of function will be described:

- **Energy monitor channel:** dropdown list for the selection of the channel (1 – 6) that will be associated to the box. The channel selected here must have been enabled from the **Energy monitor objects** screen (see section 3.2.6) under the Main configuration tab. This way, the box will be

monitoring the values received through the **[Energy x] Power**, **[Energy x] Energy**, **[Energy x] CO2** and **[Energy x] Cost estimation** objects, where “x” is the number of the assigned channel.

Important: *although it is possible to assign the “Energy monitor (KES)” function to any box from any page, it is only possible to control as many as six different energy channels, as it is not possible to have an exclusive channel for each of the boxes.*

- **Energy consumption:** determines whether the pop-up window should show the instant value of the **[Energy x] Energy** object or not.
- **CO2:** determines whether the pop-up window should show the instant value of the **[Energy x] CO2** object or not.
- **Cost:** determines whether the pop-up window should show the instant value of the **[Energy x] Cost estimation** object or not.

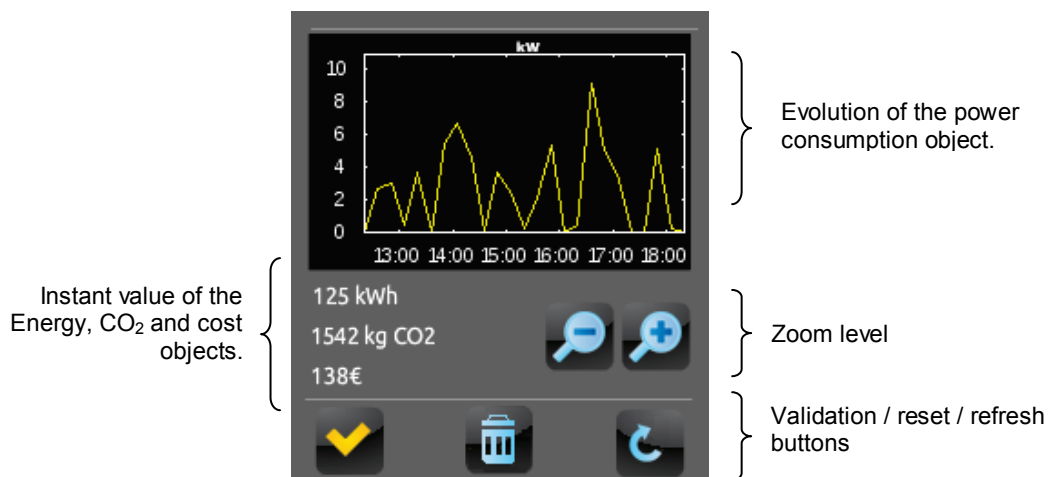


Figure 67 Energy monitor (KES). Pop-up window.

When this function is assigned to the box, two 1-bit communication objects show up: **[Pn][Bi] Energy monitor: request** and **[Pn][Bi] Energy monitor: reset**. The first of them is intended for sending a request from Z41 to KES asking for the updated values of the energy measurement, which requires having the object properly linked to the **request** object from KES, while the second object will permit sending reset orders from Z41 to KES, so that all the energy consumption, CO₂ and cost variables are set to zero, which requires having the object properly linked to the **reset** object from KES. Please refer to the user manual of KES for further information.

Note: the energy monitor objects in Z41 have no restrictions about the meaning of the data they receive from the bus, so it is entirely at the integrator's choice whether the box will monitor estimations, average consumptions, real consumptions, etc.

Finally, in the pop-up window that shows up when the box is pressed, the following elements are present:

- Evolution of the power consumption object: shows, as a graph, the chronological evolution of the value of the power object (**[Energy x] Power**). The horizontal axis will show different time lapses (hours, days, or months), depending on the selected zoom level.
- Instant value of the energy, CO₂ and cost objects: indicator that, depending on whether the **Energy consumption**, **CO2** and **Cost** parameters have been enabled, will show the current value of, respectively, the **[Energy x] Energy**, **[Energy x] CO2** and **[Energy x] Cost estimation** objects.
- Zoom level: two-button control that lets the user change the time scale, so more / less detail is shown in the graph of the evolution of the power object. The behaviour and the representation of the data are similar to those of the the numeric data log (see epigraph e) in section 3.4.2.1).
- Validation / refresh / reset buttons: permit, respectively, closing the pop-up window, sending a reset order (value "1" through the **[Pn][Bi] Energy monitor: reset** object) and requesting an updated consumption report (value "1" through the **[Pn][Bi] Energy monitor: request** object).

d) Daily timer

| | |
|---------------|-------------|
| Visualization | Other |
| Function | Daily Timer |
| Timer Type | 1-bit Value |

Figure 68 Daily timer

Boxes configured as daily timers let the final user program a timed sending of a value to the bus, being possible to send binary or scene values (according to

the parameterization) at a certain time (either everyday or only once) or at the end of a countdown.



Figure 69 Daily timer (box)

Hence, when the user presses on the box, a window containing the necessary controls for setting up a timed sending (either at a specific time or after a countdown) will pop up.

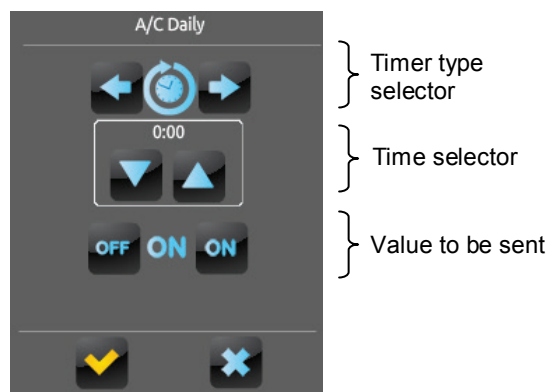





Figure 70 Daily timer (pop-up window)

This window contains the following elements:

- Timer type selector: two-button control that permits switching through the following options:

| | | | |
|---|---|--|---|
|  |  |  |  |
| Timer inactive. | Timer active (everyday). | Timer active (no repetition). | Timer active (countdown). |

- Time selector: depending on the selected timer type, permits setting the time for the automated sending, or the length of the countdown.
- Value to be sent: in case of having parameterized the sending of a binary value, this control lets the user determine the concrete value to be sent (“OFF” or “ON”). In the case of the sending of a scene, this control is not shown, as the scene number is set by parameter.

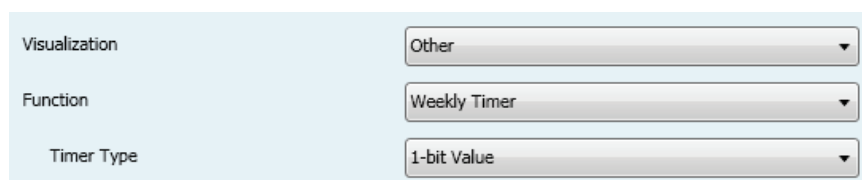
Hence, at the time set by the user or once the countdown ends, the bus will be automatically sent the corresponding value, through the **[Pn][Bi] Daily timer control** object (for binary values) or through the **[General] Scenes: send** object (in case of scene values). In both cases the **[Pn][Bi] Daily timer enabling** binary object will be available, making it possible to disable (by sending the value “0”) or to enable (value “1”) the execution of the timed sending set by the user. By default, timed sendings are always enabled.

Note: *in case of disabling a sending through **[Pn][Bi] Daily timer enabling**, the corresponding box will show the “forbidden” icon, although the box will not lose the configuration previously set by the user.*

Finally, this function offers the following parameters in ETS:

- **Timer type:** switches between “1-bit value” (default option) and “Scene”.
- **Scene number:** shown only after selecting “Scene” for the above parameter. This field defines the number of the scene (1-64) that will be sent to the bus on the execution of the timed sending.

e) Weekly timer



| | |
|---------------|--------------|
| Visualization | Other |
| Function | Weekly Timer |
| Timer Type | 1-bit Value |

Figure 71 Weekly timer

Boxes configured as weekly timers let the final user program a timed sending of a binary or scene value to the bus (according to the parameterization) at a certain time on certain days (on a weekly basis).

The difference between the daily and the weekly timers is that the latter does not permit countdown-based sendings, although it does permit that time-based sendings take place every week, on the days selected by the user.

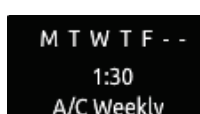


Figure 72 Weekly timer (box).

Hence, when the user presses on the box, a window similar to Figure 73 will pop up

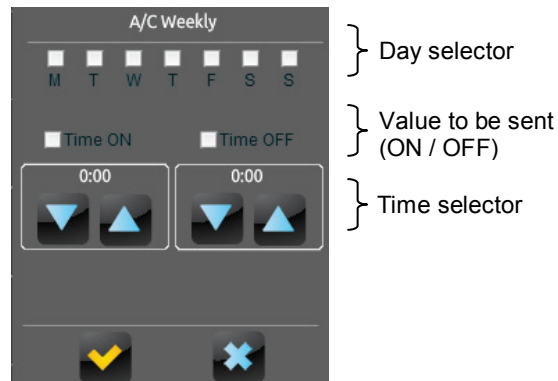


Figure 73 Weekly timer (pop-up window).

This window contains the following elements:

- Day selector: lets the user select the days on which the timed sending will take place.
- Value to be sent (ON / OFF): lets the user set the value to be sent. It can be “ON” or “OFF”, or even both at different times. One time selector is provided for each of the two values.

Note: *if instead of the sending of a binary value, the sending of a scene value has been parameterized, only one checkbox and one time selector will be shown, so the user can enable/disable the sending of the scene value and, if enabled, set the desired time.*

- Time selector: let the user set the time at which the automatic sending will take place.

Hence, at the specified time on the days selected by the user, the bus will be automatically sent the corresponding value through the **[Pn][Bi] Weekly timer control** object (in the case of sending a binary value) or through the **[General] Scenes: send** object (in the case of sending a scene number). In both cases the **[Pn][Bi] Weekly timer enabling** binary object will be available, making it possible to disable (by sending the value “0”) or to enable (value “1”) the execution of the timed sending set by the user. By default, timed sendings are always enabled.

Note: in case of disabling a sending through **[Pn][Bi] Weekly timer enabling**, the corresponding box will show the “forbidden” icon, although the box will not lose the configuration previously set by the user.

Finally, this function offers the following parameters in ETS:

- **Timer type:** switches between “1-bit value” (default option) and “Scene”.
- **Scene number:** shown only after selecting “Scene” for the above parameter. This field defines the number of the scene (1-64) that will be sent to the bus on the execution of the timed sending.

Note: the execution of user-defined timers is triggered on the detection of state transitions, even if the clock does not specifically pass through the specified time. In other words, supposing one timed sending of the value “ON” at 10:00h am and one timed sending of the value “OFF” at 8:00h pm, if the internal clock is set to 9:00h pm just after 10:00h am (and therefore after having sent the value “ON”), then the “OFF” order corresponding to 8:00h pm will take place immediately after the time change.

f) Alarm

| | |
|---------------------|-------|
| Visualization | Other |
| Function | Alarm |
| Trigger | 1 |
| Periodic Monitoring | No |

Figure 74 Alarm

Boxes configured as alarms perform user warnings on anomalous events. For this purpose, boxes of this type are related to a binary object (**[Pn][Bi] Alarm trigger**) that permits receiving alarm messages from the bus, thus making Z41 emit a continuous beep while the screen light blinks. In addition, the screen will automatically browse to the page containing the alarm box that has been triggered. After that moment, pressing on any button of the screen will interrupt the beeps and blinks although the box will still show an intermitting icon, which means that the alarm is still active and unconfirmed. Additionally, the icon of the page that contains the box will also show an exclamation.



Figure 75 Alarm (box).

The intermitting exclamation means that the alarm is active and unconfirmed.

However, alarm boxes implement an additional binary object, **[Pn][Bi] Alarm confirmation**, intended for receiving the value “1” from the bus, which will be interpreted as an acknowledgement of the alarm, thus making the icon intermittence stop. Alternatively, the alarm can be acknowledged by touching on the “OK” button in the box itself, which will cause the sending of the value “1” through the mentioned object.

The alarm will definitely become inactive once it has been confirmed and, in addition, once the **[Pn][Bi] Alarm trigger** object goes back to the “no alarm” state (the order of these two events is irrelevant), after which the icon will finally turn off.

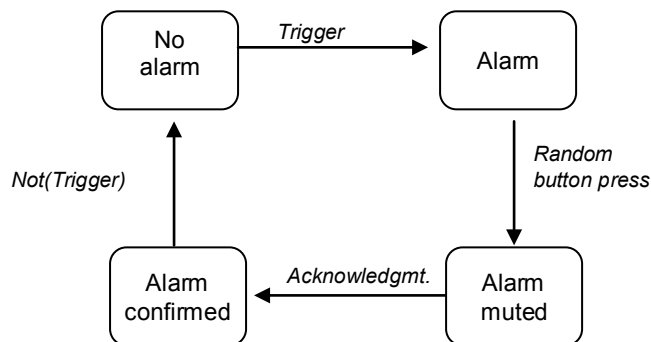


Figure 76 Alarm state diagram (case of confirmation previous to the alarm end)

There is also the possibility of **periodically monitoring** the status of the trigger object, for situations where this object is periodically received from the bus. This will let Z41 automatically assume the alarm situation if the “no alarm” value does not get sent through the trigger object after a certain time, for example upon failures of the transmitter. The maximum time window can be defined by parameter.

Consequently, alarm boxes permit configuring the following parameters:

- **Trigger:** defines the value that will trigger the alarm (“0” or “1”; it is “1” by default), i.e., the value that, when received through **[Pn][Bi] Alarm**

trigger, should be interpreted by Z41 as an alarm situation. Implicitly, this parameter also defines the inverse “no alarm” value.

- **Periodic monitoring:** activates (“Yes”) or deactivates (“No”, default option) periodic monitoring of the alarm trigger object.
- **Cycle time (x 1min.):** in case of having selected “Yes” under the above parameter, sets the maximum accepted time space without receiving the “no alarm” value before Z41 adopts the alarm situation.

Example I: *supposing a certain sensor responsible for sending the value “1” (once) to the bus when a flood is detected, and the value “0” (again, only once) when such situation terminates, if the object sent is linked to the alarm trigger object from an alarm box in Z41 where no periodical monitoring has been parameterized and where the value “1” has been set as the alarm value, then Z41 will notify an emergency every time a flood takes place. The notification will stop as soon as the user touches any button, although a warning icon will still blink on the screen. After that, if the user acknowledges the alarm, the icon will stop blinking, switching then off once the sensor sends a “0”.*

Example II: *suppose a certain CO₂ sensor responsible for sending the value “0” to the bus every two minutes, except when it detects a dangerous CO₂ level; in such case, it will immediately send the value “1”. If the object being sent is linked to the alarm trigger object from an alarm box in Z41 where periodic monitoring has been parameterized and where the value “1” is configured as the alarm trigger value, then Z41 will notify the user about the emergency both if the CO₂ level reaches a dangerous level and if the sensor stops sending the value “0”, for example due to failures or sabotage. The user should proceed in the same manner as in the above example to manage the alarm notification.*

3.5 PROFILE PAGE

The Profile Page tab contains a unique screen, Configuration.

3.5.1 CONFIGURATION

This screen permits the integrator configure the controls that will be available for the final user within the Profile page (see section 2.3), as well as the label that will identify them on screen. Note that the Profile page itself can be activated or hidden from the Configuration screen, under the Menu tab (see section 3.3.1).

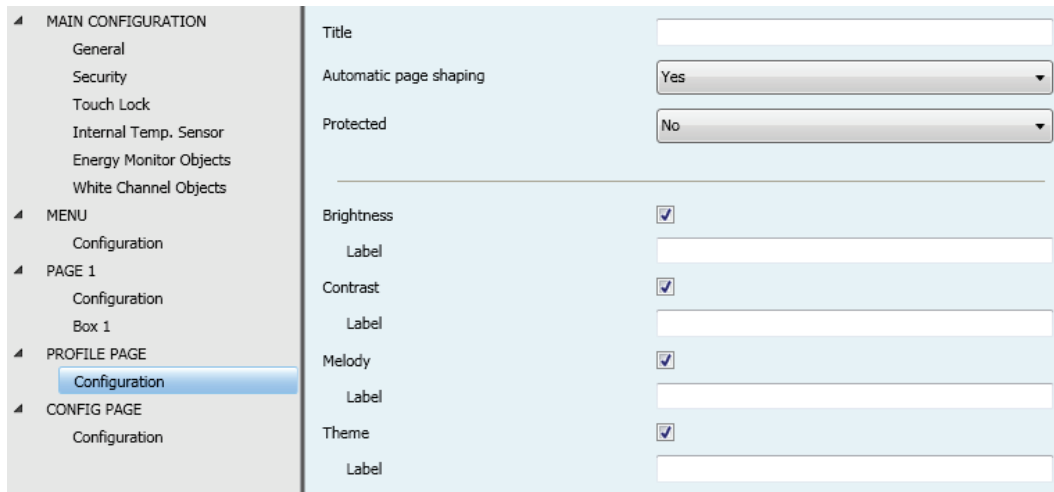


Figure 77 Configuration (Profile page)

By default, all controls are marked as active. Please refer to section 2.3 for details.

On the other hand, for the Profile page –as for any other page–, a label (parameter **Title**), a box shaping (parameter **Automatic page shaping**; see section 2.6) and a security level (parameter **Protected**; see section 3.2.2) can be set.

3.6 CONFIGURATION PAGE

The Configuration Page tab contains a unique screen, Configuration.

3.6.1 CONFIGURATION

This screen permits the integrator configure the controls that will be available for the final user within the Configuration page (see section 2.4), as well as the label that will identify them on screen. Note that the Configuration page itself can be activated or hidden from the Configuration screen, under the Menu tab (see section 3.3.1).

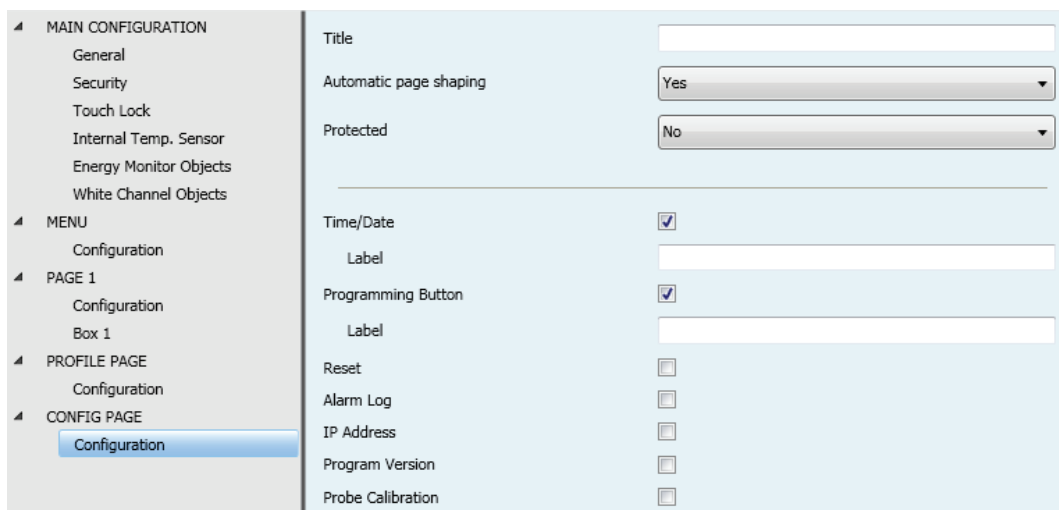


Figure 78 Configuration screen (Configuration page)

By default, the **Time/Date** and **Programming button** controls are marked as active. Please refer to section 2.4 for details.

On the other hand, for the Configuration page –as for any other page–, a label (parameter **Title**), a box shaping (parameter **Automatic page shaping**; see section 2.6), and a security level (parameter **Protected**; see section 3.2.2) can be set.

3.7 THERMOSTAT n

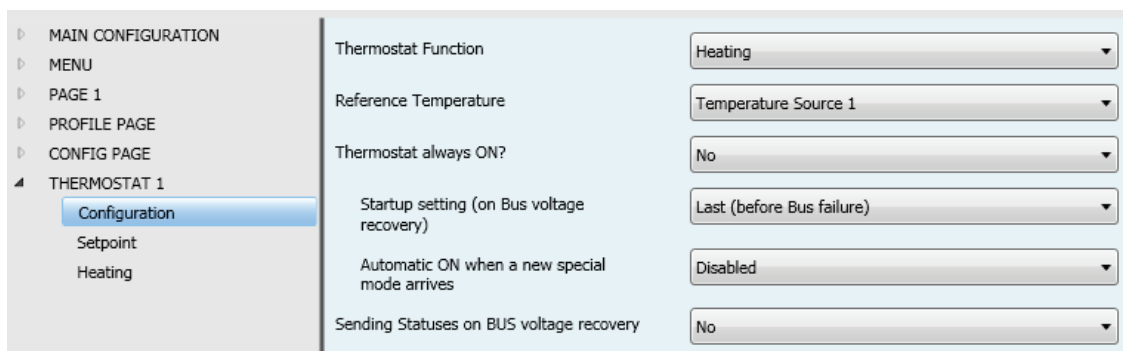


Figure 79 Configuration screen (Thermostat n)

When the functions “Thermostat 1” and “Thermostat 2” from the General screen under the Main configuration tab are enabled, the integrator will be shown two more tabs, both similar to each other, for the parameterization of these functions independently. Under each of the tabs, a set of screens (Configuration, Setpoint, etc.) is provided to define the type of the thermostatic control the user will be performing from Z41.

Note: for further information about the behaviour and the parameterization of the Zennio “Building” thermostat implemented by Z41, please refer to the specific documentation (“**Zennio Building Thermostat**”) available at www.zennio.com.

ANNEX. COMMUNICATION OBJECTS

Note: columns 1st TIME and RESET show the initial value of the object, but do not necessarily imply it is automatically sent to the bus after download or after bus failure.

| NUMBER | SIZE | I/O | FLAGS | DATA TYPE (DPT) | VALUES | | | NAME | FUNCTION |
|---|---------|-----|-----------|----------------------------|---|------------|-------|---------------------------------|---|
| | | | | | RANGE | 1ST TIME | RESET | | |
| 0 | 3 Bytes | I/O | CTRW U | [10.001] DPT_TimeOfDay | 0 | 0 | Last | [General] Time | Current Time |
| 1 | 3 Bytes | I/O | CTRW U | [11.001] DPT_Date | 01/01/1990 – 31/12/2089 | 01/01/2000 | Last | [General] Date | Current Date |
| 2 | 1 Byte | | CT--- | [18.001] DPT_SceneControl | 0-63 (run) 128-191 (learn) | - | - | [General] Scenes: send | Scene Value |
| 3 | 2 Bytes | O | CT--- | [9.001] DPT_Value_Temp | -273.00 – 670760.00 | 25 | Last | [General] Internal Temperature | Current Temperature (built-in sensor) |
| 4 | 2 Bytes | I | C--WU | [9.001] DPT_Value_Temp | -273.00 – 670760.00 | 25 | Last | [General] External Temperature | Temperature to show in the display |
| 5 | 1 Bit | I | C--W- | [1.016] DPT_Ack | 0/1 | 0 | Last | [General] Illumination | 0=No action; 1=Light Display |
| 6 | 1 Bit | I | C--W- | [1.001] DPT_Switch | 0/1 | - | - | [General] Touch Lock | 0=Touch disabled; 1=Touch enabled |
| | 1 Bit | I | C--W- | [1.001] DPT_Switch | 0/1 | - | - | [General] Touch Lock | 0=Touch enabled; 1=Touch disabled |
| 7 | 1 Bit | O | CT--- | [1.001] DPT_Switch | 0/1 | - | - | [General] Welcome Object | 1-bit Generic Control |
| 8, 11, 14, 17, 20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53, 56, 59, 62, 65, 68, 71, 74, 77, 80, 83, 86, 89, 92, 95, 98, 101, 104, 107, 110, 113, 116, 119, 122, 125, 128, 131, 134, 137, 140, 143, 146, 149 | 3 Bytes | I/O | CTRW U | [232.600] DPT_Colour_RGB | R, G, B: 0 to 255 | 00 00 00 | Last | [PX][BX] RGB Colour | Red, Green and Blue components |
| | 1 Bit | I | C--WU | [1.003] DPT_Enable | 0/1 | 0 | Last | [PX][BX] Daily Timer Enabling | 0 = Disabled; 1 = Enabled |
| | 1 Bit | I | C--WU | [1.003] DPT_Enable | 0/1 | 0 | Last | [PX][BX] Weekly Timer Enabling | 0 = Disabled; 1 = Enabled |
| | 1 Bit | I | C--WU | [1.005] DPT_Alarm | 0/1 | 0 | Last | [PX][BX] Alarm Trigger | Trigger |
| | 1 Byte | I/O | CTRW U | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Red Channel | 0-100% |
| | 1 Byte | I | C--WU | [20.102] DPT_HVACMode | 0=Confort; 1=Standby; 3=Economy; 4=Building Protection & Rest=Nothing | 1 | Last | [PX][BX] Special Mode Indicator | Comfort, Standby, Economy and Building Protection |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Fan Indicator | 0=Off; 1-33%=Min; 34-66%=Med; >66%=Max |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Fan Indicator | 0=Auto; 1-33%=Min; 34-66%=Med; >66%=Max |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Fan Indicator | 0-33%=Min; 34-66%=Med; >66%=Max |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Fan Indicator | 0-49%=Min; 50-100%=Max |
| | 1 Byte | I | C--WU | [20.105] DPT_HVACContrMode | 1=Heat; 3=Cool; 0=Auto; 9=Fan; 14=Dry & rest=Nothing | 0 | Last | [PX][BX] Mode Indicator | Heat, Cool, Auto, Fan and Dry |

| NUMBER | SIZE | I/O | FLAGS | DATA TYPE (DPT) | VALUES | | | NAME | FUNCTION |
|---------|--|-------|----------------------------|----------------------------|---|----------|-------------------------------|--|---|
| | | | | | RANGE | 1ST TIME | RESET | | |
| | 1 Bit | I | C--WU | [1.100] DPT_Heat_Cool | 0/1 | 0 | Last | [PX][BX] Mode Indicator | 0 = Cool; 1 = Heat |
| | 2 Bytes | I | C--WU | [9.001] DPT_Value_Temp | -20.00 – 95.00 | 25 | Last | [PX][BX] Temperature Indicator | Absolute Float Value |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Light Indicator | 0%=Off; 100%=On |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0 – 100 | 0 | Last | [PX][BX] Shutter Position | 0%=Top; 100%=Bottom |
| | 2 Bytes | I | C--WU | [9.xxx] | -671088.64 – 670760.96 | 0 | Last | [PX][BX] 2-byte Float Indicator | -671088.64...670760.96 |
| | 2 Bytes | I | C--WU | [8.001] DPT_Value_2_Count | -32768 – 32767 | 0 | Last | [PX][BX] 2-byte Signed Int Indicator | -32768...32767 |
| | 2 Bytes | I | C--WU | [7.001] DPT_Value_2_Ucount | 0 -65535 | 0 | Last | [PX][BX] 2-byte Unsigned Int Indicator | 0...65535 |
| | 1 Byte | I | C--WU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Percentage Indicator | 0-100% |
| | 1 Byte | I | C--WU | [6.010] DPT_Value_1_Count | -128 – 127 | 0 | Last | [PX][BX] 1-byte Signed Int Indicator | -128...127 |
| | 1 Byte | I | C--WU | [5.010] DPT_Value_1_Ucount | 0 – 255 | 0 | Last | [PX][BX] 1-byte Unsigned Int Indicator | 0...255 |
| | 1 Byte | I | C--WU | [5.010] DPT_Value_1_Ucount | 0 – 255 | 0 | Last | [PX][BX] Enumerated Indicator | 0...255 |
| | 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99, 102, 105, 108, 111, 114, 117, 120, 123, 126, 129, 132, 135, 138, 141, 144, 147, 150 | 1 Bit | I | C--WU | [1.001] DPT_Switch | 0/1 | 0 | Last | [PX][BX] Binary Indicator |
| 1 Bit | | O | CTR-- | [1.017] DPT_Trigger | 0/1 | - | - | [PX][BX] Energy Monitor: request | 0=No Action; 1=Request |
| 1 Bit | | I | CT-W- | [1.001] DPT_Switch | 0/1 | 0 | - | [PX][BX] Alarm Confirmation | 0=No Action; 1=Cofirm |
| 1 Bit | | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [PX][BX] Weekly Timer Control | 1-bit Control |
| 1 Bit | | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [PX][BX] Daily Timer Control | 1-bit Control |
| 1 Byte | | I/O | CTRWU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Green Channel | 0-100% |
| 1 Byte | | O | CTR-- | [20.102] DPT_HVACMode | 0=Confort; 1=Standby; 3=Economy; 4=Building Protection | 0 | Last | [PX][BX] Special Mode Control | Comfort, Standby, Economy and Building Protection |
| 1 Bit | | O | CTR-- | [1.007] DPT_Step | 0 | 0 | Last | [PX][BX] Fan Control | 0=Decrease; 1=Increase |
| 1 Byte | | O | CTR-- | [20.105] DPT_HVACContrMode | 1=Heat; 3=Cool; 0=Auto; 9=Fan; 14=Dry | 0 | Last | [PX][BX] Mode Control | Heat, Cool, Auto, Fan and Dry |
| 1 Bit | | O | CTR-- | [1.100] DPT_Heat_Cool | 0/1 | 0 | Last | [PX][BX] Mode Control | 0 = Cool; 1 = Heat |
| 1 Bit | | O | CTR-- | [1.007] DPT_Step | 0/1 | 0 | Last | [PX][BX] Temperature Offset Control | 0=Decrease; 1=Increase |
| 2 Bytes | | I/O | CTRW- | [9.002] DPT_Value_Tempd | -10 – 10 | 0 | Last | [PX][BX] Temperature Offset Control | Relative Float Value |
| 2 Bytes | | O | CTR-- | [9.001] DPT_Value_Temp | -20.00 – 95.00 | 25 | Last | [PX][BX] Temperature Control | Absolute Float Value |
| 1 Bit | | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [PX][BX] Light On/Off | 0=Off; 1=On |
| 1 Bit | | O | CTR-- | [1.008] [1.008] DPT_UpDown | 0/1 | 0 | Last | [PX][BX] Move Shutter | 0=Up; 1=Down |
| 1 Byte | O | CTR-- | [5.010] DPT_Value_1_Ucount | 0-255 | 0 | Last | [PX][BX] Enumerated Control | 0...255 | |
| 2 Bytes | O | CTR-- | [9.xxx] | -671088.64 – 670760.96 | 0 | Last | [PX][BX] 2-byte Float Control | -671088.64...670760.96 | |

| NUMBER | SIZE | I/O | FLAGS | DATA TYPE (DPT) | VALUES | | | NAME | FUNCTION |
|--|---------|-----|-------|-----------------------------|---|----------|-------|--------------------------------------|----------------------------|
| | | | | | RANGE | 1ST TIME | RESET | | |
| | 2 Bytes | O | CTR-- | [8.001] DPT_Value_2_Count | -32768 – 32767 | 0 | Last | [PX][BX] 2-byte Signed Int Control | -32768...32767 |
| | 2 Bytes | O | CTR-- | [7.001] DPT_Value_2_Ucount | 0 – 65535 | 0 | Last | [PX][BX] 2-byte Unsigned Int Control | 0...65535 |
| | 1 Byte | O | CTR-- | [5.001] DPT_Scaling | 0 – 100 | 0 | Last | [PX][BX] Percentage Control | 0-100% |
| | 1 Byte | O | CTR-- | [6.010] DPT_Value_1_Count | -128 – 127 | 0 | Last | [PX][BX] 1-byte Signed Int Control | -128...127 |
| | 1 Byte | O | CTR-- | [5.010] DPT_Value_1_Ucount | 0 – 255 | 0 | Last | [PX][BX] 1-byte Unsigned Int Control | 0...255 |
| | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [PX][BX] Binary Control | 1-bit Control |
| 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43, 46, 49, 52, 55, 58, 61, 64, 67, 70, 73, 76, 79, 82, 85, 88, 91, 94, 97, 100, 103, 106, 109, 112, 115, 118, 121, 124, 127, 130, 133, 136, 139, 142, 145, 148, 151 | 1 Bit | O | CTR-- | [1.015] DPT_Reset | 0/1 | 0 | Last | [PX][BX] Energy Monitor: reset | 0=No Action; 1=Reset |
| | 1 Byte | I/O | CTRWU | [5.001] DPT_Scaling | 0-100 | 0 | Last | [PX][BX] Blue Channel | 0-100% |
| | 4 Bit | O | CTR-- | [3.007] DPT_Control_Dimming | 0-100 | 0 | Last | [PX][BX] Light Dimming | 4-bit Dimming Control |
| | 1 Bit | O | CTR-- | [1.007] DPT_Step | 0/1 | 0 | Last | [PX][BX] Stop Shutter | 0 or 1 -> Stop |
| 152, 179 | 2 Bytes | I | ---W- | [9.001] DPT_Value_Temp | -273.00 – 670760.00 | 25 | Last | [TX] Temperature Source 1 | External Sensor Measure |
| 153, 180 | 2 Bytes | I | ---W- | [9.001] DPT_Value_Temp | -273.00 – 670760.00 | 25 | Last | [TX] Temperature Source 2 | External Sensor Measure |
| 154, 181 | 1 Byte | I | ---W- | [20.102] DPT_HVACMode | 0=Confort; 1=Standby; 3=Economy; 4=Building Protection | 0 | Last | [TX] Special Mode | 1-byte HVAC Mode |
| 155, 182 | 1 Bit | I | ---W- | [1.001] DPT_Switch | 0/1 | Paramet. | Last | [TX] Special Mode: comfort | 0=Off; 1=On |
| | 1 Bit | I | ---W- | [1.017] DPT_Trigger | 0/1 | Paramet. | Last | [TX] Special Mode: comfort | 0=Nothing; 1=Trigger |
| 156, 183 | 1 Bit | I | ---W- | [1.001] DPT_Switch | 0/1 | Paramet. | Last | [TX] Special Mode: standby | 0=Off; 1=On |
| | 1 Bit | I | ---W- | [1.017] DPT_Trigger | 0/1 | Paramet. | Last | [TX] Special Mode: standby | 0=Nothing; 1=Trigger |
| 157, 184 | 1 Bit | I | ---W- | [1.001] DPT_Switch | 0/1 | Paramet. | Last | [TX] Special Mode: economy | 0=Off; 1=On |
| | 1 Bit | I | ---W- | [1.017] DPT_Trigger | 0/1 | Paramet. | Last | [TX] Special Mode: economy | 0=Nothing; 1=Trigger |
| 158, 185 | 1 Bit | I | ---W- | [1.001] DPT_Switch | 0/1 | 0 | Last | [TX] Special Mode: protection | 0=Off; 1=On |
| | 1 Bit | I | ---W- | [1.017] DPT_Trigger | 0/1 | 0 | Last | [TX] Special Mode: protection | 0=Nothing; 1=Trigger |
| 159, 186 | 1 Bit | I | ---W- | [1.019] DPT_Window_Door | 0/1 | 0 | Last | [TX] Window Status (input) | 0=Closed; 1=Open |
| 160, 187 | 1 Bit | I | ---W- | [1.017] DPT_Trigger | 0/1 | 0 | Last | [TX] Comfort Prolongation | 0=Nothing; 1=Timed Comfort |
| 161, 188 | 1 Byte | O | CTR-- | [20.102] DPT_HVACMode | 0=Confort; 1=Standby; 3=Economy; 4=Building Protection | 0 | Last | [TX] Special Mode Status | 1-byte HVAC Mode |
| 162,189 | 2 Bytes | I | ---W- | [9.001] DPT_Value_Temp | 5.00 – 45.00 | Paramet. | Last | [TX] Basic Setpoint | Reference setpoint |
| | 2 Bytes | I | ---W- | [9.001] DPT_Value_Temp | 5.00 – 45.00 | Paramet. | Last | [TX] Setpoint | Thermostat setpoint input |
| 163, 190 | 1 Bit | I | ---W- | [1.007] DPT_Step | 0/1 | 0 | Last | [TX] Setpoint Step | 0=-0.5°C; 1=+0.5°C |
| 164, 191 | 2 Bytes | I | ---W- | [9.002] DPT_Value_Tempd | -10.00 – 10.00 | Paramet. | Last | [TX] Setpoint Offset | Float offset value |
| 165, 191 | 2 Bytes | O | CTR-- | [9.001] DPT_Value_Temp | 5.00 – 45.00 | Paramet. | Last | [TX] Setpoint Status | Current setpoint |

| NUMBER | SIZE | I/O | FLAGS | DATA TYPE (DPT) | VALUES | | | NAME | FUNCTION |
|------------------------------|---------|-----|-------|--|--------------------------|----------|----------|------------------------------|--------------------------------|
| | | | | | RANGE | 1ST TIME | RESET | | |
| 166, 193 | 2 Bytes | O | CTR-- | [9.001] DPT_Value_Temp | 5.00 – 45.00 | Paramet. | Last | [TX] Basic Setpoint Status | Current basic setpoint |
| 167, 194 | 2 Bytes | O | CTR-- | [9.002] DPT_Value_Tempd | -10.00 – 10.00 | Paramet. | Last | [TX] Setpoint Offset Status | Current setpoint offset |
| 168, 195 | 1 Bit | I | ---W- | [1.015] DPT_Reset | 0/1 | 0 | Last | [TX] Offset Reset | Reset offset |
| | 1 Bit | I | ---W- | [1.015] DPT_Reset | 0/1 | 0 | Last | [TX] Setpoint Reset | Reset setpoint to default |
| 169, 196 | 1 Bit | I | ---W- | [1.100] DPT_Heat_Cool | 0/1 | Paramet. | Last | [TX] Mode | 0 = Cool; 1 = Heat |
| 170, 197 | 1 Bit | O | CTR-- | [1.100] DPT_Heat_Cool | 0/1 | Paramet. | Last | [TX] Mode Status | 0 = Cool; 1 = Heat |
| 171, 198 | 1 Bit | I | ---W- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] On/Off | 0=Off; 1=On |
| 172, 199 | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] On/Off Status | 0=Off; 1=On |
| 173, 200 | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Cool) | PI Control (PWM) |
| | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Cool) | 2-Point Control |
| 174, 201 | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Heat) | PI Control (PWM) |
| | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Heat) | 2-Point Control |
| 175, 202 | 1 Byte | O | CTR-- | [5.001] DPT_Scaling | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Cool) | PI Control (Continuous) |
| 176, 203 | 1 Byte | O | CTR-- | [5.001] DPT_Scaling | 0/1 | Paramet. | Paramet. | [TX] Control Variable (Heat) | PI Control (Continuous) |
| 177, 204 | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [TX] Additional Cool | Temp >= (Setpoint+Band) => "1" |
| 178, 205 | 1 Bit | O | CTR-- | [1.001] DPT_Switch | 0/1 | 0 | Last | [TX] Additional Heat | Temp <= (Setpoint-Band) => "1" |
| 206, 210, 214, 218, 222, 226 | 2 Bytes | I | ---W- | [9.024] DPT_Power | -670760,00 – 670760,00 | 0 | Last | [Energy X] Power | kW |
| 207, 211, 215, 219, 223, 227 | 4 Bytes | I | ---W- | [13.013] DPT_ActiveEnergy_kWh | -2147483648 – 2147483647 | 0 | Last | [Energy X] Energy | kWh |
| 208, 212, 216, 220, 224, 228 | 2 Bytes | I | ---W- | [9.xxx] CO2 emission non-standardized datapoint type | -670760,00 – 670760,00 | 0 | Last | [Energy X] CO2 | kg CO2 |
| 209, 213, 217, 221, 225, 229 | 2 Bytes | I | ---W- | [9.xxx] Currency non-standardized datapoint type | -670760,00 – 670760,00 | 0 | Last | [Energy X] Cost Estimation | Local Currency |
| 230, 231, 232, 233 | 1 Byte | O | CTR-- | [5.001] DPT_Scaling | -670760,00 – 670760,00 | 0 | Last | [RGBW] White Channel X | 0-100% |

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