Binary Inputs

Input Module for Push Buttons, Switches and On/Off Sensors.

USER MANUAL

Software Library Version: [1.0]
User Manual Version: [1.0]_a

www.zennio.com

CONTENTS

DOCUMENT UPDATES

Version	Changes	Page(s)
[1.0]_a	**Changes in the software library**: • Internal code optimisation. • Minor revision of parameter texts.	-
	Reorganisation of the document structure.	-
[0.3]_a	**Changes in the software library**: • Debounce time. • Pulse counter (only on BIN devices). • Double press for push button (only on BIN devices).	-
[0.2]_a	**Changes in the software library** : • New 'Delay' option for all push button actions (shutter, dimmer, scene, constants). • Minor revision of parameter texts.	-

1 INTRODUCTION

A variety of Zennio devices incorporate an input interface where it is possible to connect one or more **push buttons**, **switches** or **on-off sensors**, among other accessories.

In addition, the binary inputs module of **BIN family devices from Zennio** also implements pulse counting capabilities as well as some additional features, such as double press detection for push buttons.

Please refer to the specific user manual and datasheet of the Zennio device in order to confirm whether these features are available or not, and for instructions on how to connect these accessories to the input interface of the device.

On the other hand, keep in mind that even if the input accessory is the same in all cases, **the functionality and the ETS configuration may slightly differ depending on the device it is being connected to and the version of its application program**. Please always ensure to download from the Zennio homepage (www.zennio.com) the user manual and annexes that correspond to the specific device and application program being configured.

2 CONFIGURATION

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

2.1 CONFIGURATION

In the "Configuration" tab, the functionality related to **binary inputs** is possible to enable.

In addition, a **debounce time** can be set, so bounces in the input signal are ignored within a certain period. When the pushbuttons or switches introduce bounces in the signal due to mechanical effects, this functionality may be useful to prevent undesired behaviours, such as inconsistence in the pulse count or wrong detection of edges and button presses.

ETS PARAMETERISATION

The application program will typically provide a box for each input, so that configuring them as a binary input independently is possible. Please refer to the application program specific manual to identify where these boxes are located.

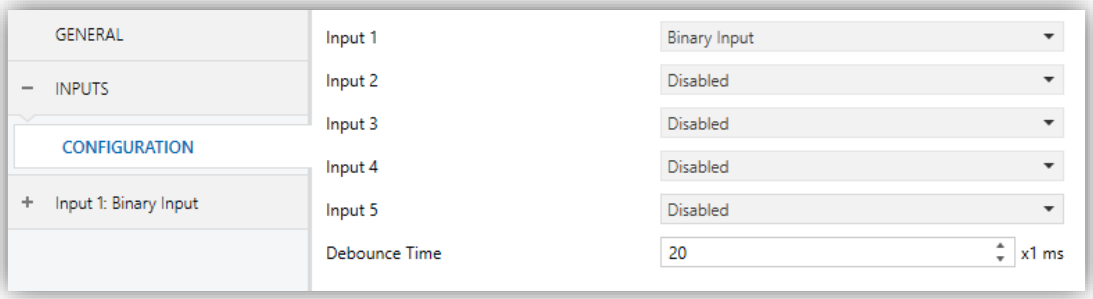

Figure 1. Enabling the binary input module.

- **Input X:** enables the "Input X: binary input" tab in the left menu (see section 2.2).

- **Debounce Time** [*10…20…255*][1] [*x1 ms*]: time interval after a rising edge or a falling edge during which any further pulses in the input are ignored.

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

2.2 INPUT X: BINARY INPUT

Inputs configured as binary inputs let the device perform the following tasks:

- Retrieving the **state** (1/0) of the input line and detecting **changes** (i.e., button presses, sensor changes, etc.).
- **Reporting** the KNX bus about the above states/changes and triggering the corresponding **actions**, depending on the case.
- Detecting **sabotage** (i.e., unexpected voltage levels on the line) on inputs configured as switch/sensor. This function is not available on BIN devices.

Every binary input needs itself to be configured as a **push button** or a **switch/sensor**. Additionally, BIN devices allow connecting pulse generators and therefore configure the inputs as **pulse counters**.

It is possible to **lock/unlock** each input independently by writing to the proper objects. While an input remains locked, the application will ignore further switches that may take place on the line, however the **periodical sending** of values, if parameterised, will not be interrupted (the last value will still be re-sent, even if the input switches the state). On the other hand, when the **unlock** event occurs, a fresh evaluation of the current state of the input will be performed. If it differs from the state prior to the lock, it will be assumed that an edge has taken place and, therefore, the associated action will be activated.

ETS PARAMETERISATION

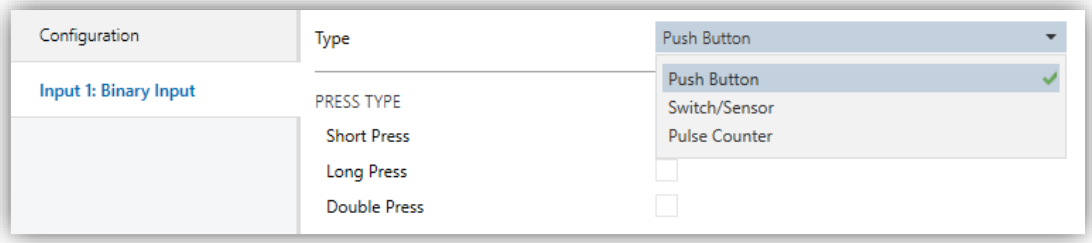

Figure 2. Binary input – Configuration

- **Type** [*Push Button / Switch/Sensor / Pulse Counter*]: sets whether the input is a "Push button" (see section 2.2.1), a "Switch/sensor" (see section 2.2.2) or, in the case of the BIN devices, as a "Pulse Counter" (see section 2.2.3).

In addition, the "**[Ix] Input Lock**" object turns visible. When it receives a "1", the input will become locked, while a "0" will unlock it.

2.2.1 PUSH BUTTON

Note: *double press detection is only available on BIN devices*.

The actions to be triggered on **short**, **double** and **long** presses (and even on the release of the push button) are independent and parameterisable. It is even possible to set how long a press must be to be considered as long or how much time must elapse between two consecutive short presses to be considered as a double press.

When configuring a double press, these actions can consist in **sending the KNX bus** a binary value or a scene run/save order, while for short and long presses it is also possible to send a shutter control order, a dimmer control order, or a constant numeric value.

For every type of press, a certain **delay** before sending the value to the bus can be configured in some cases. In the case of the binary values, a **periodical re-sending** can be also configured, which may be useful if intending to send such value to an alarm monitor or similar.

ETS PARAMETERISATION

When "Push button" is selected in Binary Input **Type**, these parameters are available:

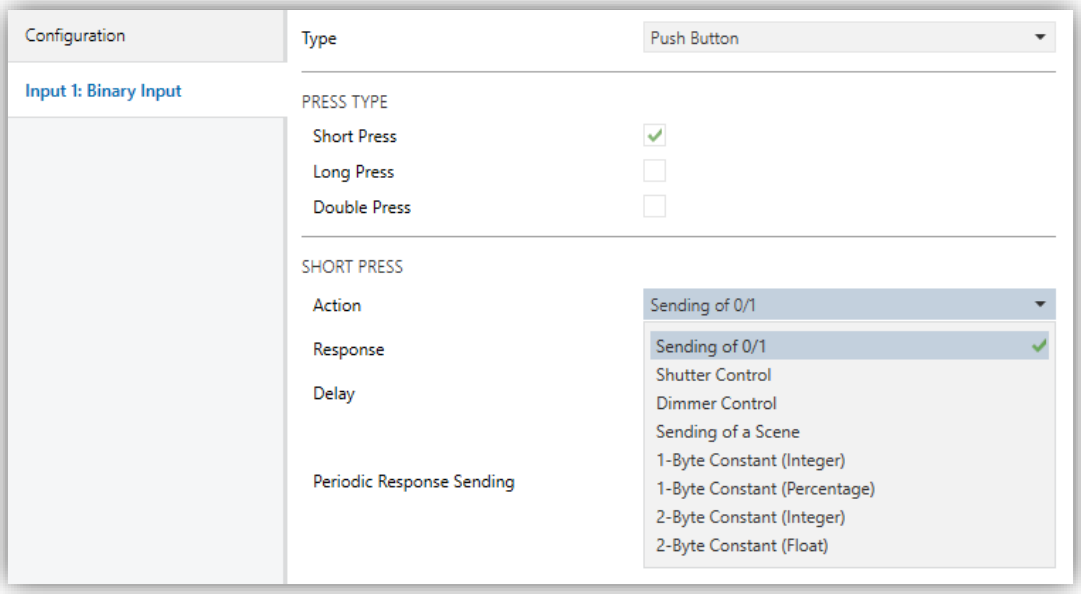

Figure 3. Push Button (**BIN devices**)

- **Short Press**.
- **Long Press**. If enabled, the following parameter appears:
  - **Long Press Time** [*1…5…50*] [*x0,1 s*]: sets the minimum time that a press should last to consider it a long press.
- **Double Press** (only on BIN devices). If enabled, the following parameter appears:
  - **Double Press Time** [*1…5…50*] [*x0,1 s*]: sets the maximum time between the two button presses to consider them a double press.

    Note: *the greater this time, the more the device will need to wait after a single press to make double presses possible. To prevent delaying the reactions to single presses significantly, values lower than 0.5 seconds are advisable for this parameter.*

Note: *Long Press and Double Press cannot be enabled simultaneously.*

Short Press

- **Action**: action to be triggered when a short press is detected:
  - *Sending of 0/1*.
    - **Response** [*0 / 1 / Toggle 0/1*]: sets the value "B" to be sent through object "**[Ix] [Short Press] B**".
    - **Periodical Response Sending** [*No / Only for value 0 / Only for value 1 / Always*]: sets whether to re-send the response values periodically or not.
      - **Sending Period** [*1…255*][*s/min*] [*1…18*][*h*]: the cycle time.
  - *Shutter Control*.
    - **Response**: sets the particular order to be sent to the shutter actuator:
      - *Up*: one "0" will be sent through "**[Ix] [Short Press] Move Up Shutter**",
      - *Down*: one "1" will be sent through "**[Ix] [Short Press] Move Down Shutter**",

- *Up/Down (switched)*: values "1" and "0" will be sent alternatively through "**[Ix] [Short Press] Move Up/Down Shutter**". In this case an additional, writable object ("**[Ix] [Short Press] Shutter Status (input)**") will be available; it may be linked to the shutter status object from the shutter actuator to receive feedback about the current state of the shutter. This will avoid sending move-up orders if the shutter is already at 0% or move-down orders if it is already at 100%.
- *Stop/Step Up*: one "0" will be sent through "**[Ix] [Short Press] Stop/Step Up Shutter**",
- *Stop/Step Down*: one "1" will be sent through "**[Ix] [Short Press] Stop/Step Down Shutter**",
- *Stop/Switched Step*: values "1" and "0" will be sent (alternatively with every press) through "**[Ix] [Short Press] Stop/Step Shutter (switched)**".

➢ *Dimmer Control*.

- **Response**: sets the particular order to be sent to the light dimmer:
  - *Light On*: one "1" will be sent through the "**[Ix] [Short Press] Light On**" binary object,
  - *Light Off*: one "0" will be sent through the "**[Ix] [Short Press] Light Off**" binary object,
  - *Light On/Off (switched)*: values "1" and "0" will be sent alternatively through the "**[Ix] [Short Press] Light On/Off**" binary object,
  - *Brighter*: on every odd press, one 4-bit order will be sent (through "**[Ix] [Short Press] Brighter**") to increase the light level by a certain percentage, which needs to be configured through parameter "**Dimming Step**". On the other hand, on every even press, an order to stop the dimming will be sent. The sequence is therefore Brighter → Stop → Brighter → Stop → etc.
  - *Darker*: analogous to the previous one, but for decreasing the light level,
  - *Brighter/Darker (Switched)*: analogous to the above two, although the dimming orders will be in this case increase/decrease orders

(alternatively). The sequence will be therefore Brighter → Stop → Darker → Stop → Brighter → etc. These orders will be sent through object "**[Ix] [Short Press] Brighter/Darker**". In this case there will be an additional, writable object ("**[Ix] [Short Press] Dimming Status (Input)**" that may be linked to the dimming status object from the dimmer, in order to receive feedback about the current light level. This will avoid increase orders if the current level is 100% or decrease orders if it is already 0%

- *Sending of a Scene*.
  - **Response** [*Run Scene / Save Scene*]: sets the particular order to be sent to the bus. If "Run Scene" is selected, an order to run a specific scene will be sent through "**[Ix] [Short Press] Run Scene**". If "Save Scene" is selected, an order to save the current state as a specific scene will be sent through "**[Ix] [Short Press] Save Scene**".
  - **Scene** [*1…64*]: sets the desired scene number for the above run/save orders.
- *1-Byte Constant (Integer)*:
  - **Response** [*0…255*]: sets a constant value to be sent to the bus (through "**[Ix] [Short Press] Constant Value (Integer)**").
- *1-Byte Constant (Percentage)*:
  - **Response** [*0…100*]: sets a constant value to be sent to the bus (through "**[Ix] [Short Press] Constant Value (Percentage)**").
- *2-Byte Constant (Integer)*:
  - **Response** [*0…65535*]: sets a constant value to be sent to the bus (through "**[Ix] [Short Press] Constant Value (Integer)**").
- *2-Byte Constant (Float)*:
  - **Response** [*-671088,64…0…670760,96*]: sets a constant value to be sent to the bus (through "**[Ix] [Short Press] Constant Value (float)**").

- **Delay** [*0…255*][*s/min*] [*0…18*][*h*]: sets a delay between the detection of the button press and the actual sending of the response.

Long Press

- **Action**: defines the action to be triggered when a long press is detected. The available actions are **analogous to those for the short press**, except for the following remarks:

  - In the shutter control, if a move-up or move-down (or an alternating move up/down) order is configured as the response, then together with the usual object through which the parameterised order is sent after pressing the button, an additional object will show up: "**[Ix] [Long Press] Stop Shutter**", which will send an order to stop the motion of the shutter as soon as the button is released, thus making it possible to (optionally) implement a hold & release shutter control.

    ***Example**: Long-Press Shutter Control.*

    *"Shutter Control" has been set as the desired reaction to long presses, and "Move up" as the response. When a long press is detected, the value "0" will be sent through "**[Ix] [Long Press] Move up Shutter**", while after the release of the push button the value "0" will be sent through "**[Ix] [Long Press] Stop Shutter**", which will only take effect if it gets linked to the corresponding object from the shutter actuator.*

  - In the dimmer control, if an order to increase or decrease the light level (or to increase / decrease it alternatively) is configured as the response, then the usual object through which the parameterised order is sent after pressing the button **will also send an order to stop the light regulation** as soon as the button is released (thus permitting a hold & release dimmer control).

    ***Example**: Long-Press Dimmer Control.*

    *"Dimmer Control" has been set as the desired reaction to long presses, and "Brighter" (with a dimming step of 50%) as the response. When a long press is detected, the value "0xA" will be sent through "**[Ix] [Long Press] Brighter**", while after the release of the push button the value "0x8" will be sent, which will interrupt the light regulation.*

Double Press

Note: *only available on BIN devices.*

- **Action** [*Sending of 0/1 / Sending of a Scene*]: defines the action to be triggered when a double press is detected. The functionality of the available actions is analogous to that described for Short Press and Long Press.

2.2.2 SWITCH/SENSOR

Binary values (configurable) will be sent to the bus whenever rising or falling edges are detected in the input line.

It is possible to introduce a certain **delay** prior to sending these values to the KNX bus – a delay for the "0" and a delay for the "1", no matter which one is sent after which edge (rising / falling). Moreover, **periodically re-sending** the last value is possible by configuring the desired period.

Security checks can optionally be performed for inputs of the Switch/Sensor type, as long as an **end-of-line resistor** has been connected to the switch/sensor. The value of such resistor needs to be configured by parameter (the available values are 2.2 kΩ, 2.7 kΩ, 3.3 kΩ, 4.7 kΩ and 10 kΩ) as well as whether it has been connected in parallel or serially, which depends on the switch/sensor type (*normally open* or *normally closed*).

- If **normally open**, the line will remain at a low voltage level in the absence of the undesired condition. The occurrence of that situation, however, will cause a rising edge (the switch/sensor gets closed). This type of sensor requires connecting the end-of-line resistor **in parallel**.

- On the other hand, if **normally closed**, the line remains at a high voltage level until the occurrence of the undesired situation, which will cause a falling edge (the switch/sensor will open). This requires connecting the end-of-line resistor **in series**.

By means of this resistor, it will be possible to distinguish not only the two states of the switch/sensor, but also additional (unexpected) voltage levels (e.g., short-circuits and open circuits due to a **breakdown** or a **sabotage**), which will be reported to the bus through alarm objects.

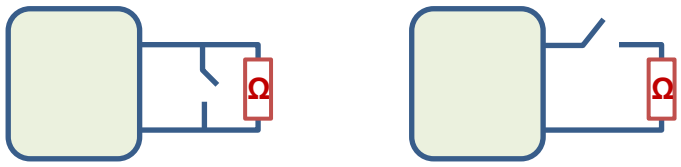

Figure 4. Left: normally open Sensor (parallel resistor). Right: Normally Closed Sensor (serial resistor).

Note: *security functionality is not available on BIN devices*.

ETS PARAMETERISATION

When "Switch/Sensor" is selected in Binary Input **Type**, the following parameters are available:

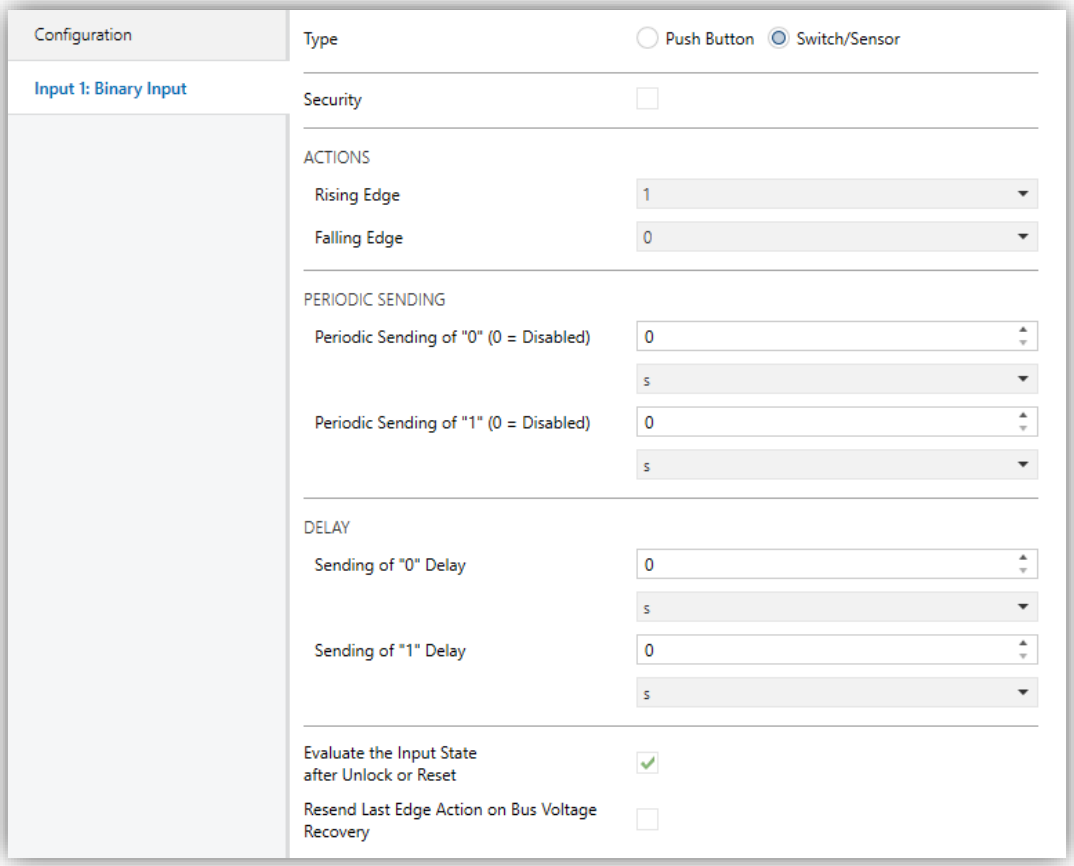

Figure 5. Switch/Sensor

- **Security** [*disabled*/*enabled*] (option not available on BIN devices): selecting or unselecting this checkbox determines whether the input line includes a security end-of-line resistor so it is therefore possible to monitor sabotage or breakdown situations (which will be notified by periodically sending the value "1" through object "**[Ix] [Switch/Sensor] Alarm: Breakdown or Sabotage**";

once they are over, this object will send one "0"). When selected, two more parameters show up:

- **Switch/Sensor Type** [*N.O. (Parallel Resistor) / N.C. (Serial Resistor)*]: sets whether the switch/sensor is normally open and therefore with a resistor connected in parallel or normally closed and therefore with a resistor connected in series.

- **Resistor Value** [*2.2 kΩ / 2.7 kΩ / 3.3 kΩ / 4.7 kΩ / 10 kΩ*]: sets the value of the resistor.

- **Actions**.

  - **Rising Edge** [*No Action / 0 / 1 / Switching 0/1*]: indicates what should be sent to the KNX bus when a rising edge takes place on the input line. The values are sent through object "**[Ix] [Switch/Sensor] Edge**".

  - **Falling Edge** [*No Action / 0 / 1 / Switching 0/1*]: analogous to the above parameter. The response to falling edges will be sent through the same object ("**[Ix] [Switch/Sensor] Edge**").

- **Periodical Sending**.

  - **Sending of "0"** [*0...255*][*s/min*] [*0…18*][*h*]: sets every how much time the value "0" should be periodically sent once the corresponding edge has been detected. If no periodical re-sending is required, just leave this at 0.

  - **Sending of "1"** [*0...255*][*s/min*] [*0…18*][*h*]: analogous to the above one, but for the value "1".

- **Delay**.

  - **Sending of "0"** [*0...255*][*s/min*] [*0…18*][*h*]: sets a certain delay before sending the value "0" once the corresponding edge has been detected. For an immediate sending, just leave this at 0.

  - **Sending of "1"** [*0...255*][*s/min*] [*0…18*][*h*]: analogous to the above one, but for the value "1".

- **Evaluate the Input State after Unlock or Reset** [*disabled/enabled*]: sets whether the state of the input line should be evaluated or not whenever it gets

unlocked or when the device recovers from a power loss, so this new state can be compared to the last known, making the device trigger the proper response in case the two are different.

- **Resend Last Edge Action on Bus Voltage Recovery** [*disabled/enabled*]: sets if the status of the line (i.e., the action that corresponds to the fact of switching to that state) should always be sent to the bus when the device recovers from a power loss, even if the status is the same as before the power loss.

2.2.3 PULSE COUNTER

Note: *this functionality is only available on BIN devices*.

The pulse counter function consists in counting the number of edges detected in the input. It is possible to select the **edge type** to be counted and the **counter size**. Moreover, depending on the configuration, the sending to the KNX bus can take place **periodically**, when the **value changes** or once a certain **target value** is reached.

The current value of the counter can be **reset to zero** any time (provided that the input is not locked) through a specific binary object.

ETS PARAMETERISATION

When "Pulse Counter" is selected in Binary Input **Type**, the following parameters are available:

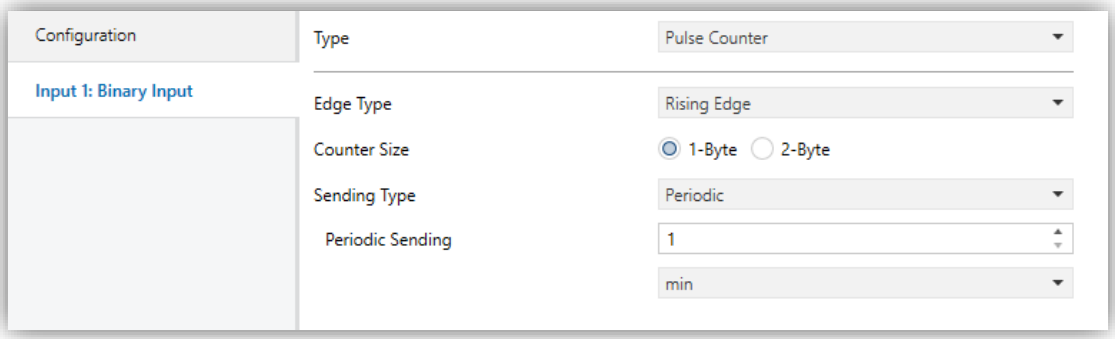

Figure 6. Pulse Counter.

- **Edge Type** [*Rising Edge / Falling Edge / Rising and Falling Edge*]: sets the event type that must update the counter value.

- **Counter Size** [*1-Byte / 2-Byte*]: sets the counter size, which determines the maximum value of the counter (1-255 for "1 byte"; 1-65535 for "2-byte").

- **Sending Type**: sets when the counter value should be sent to the bus, through the "**[Ix] [Pulse Counter] Counter**" object.

  - *Periodic*: the object will be sent cyclically. The period must be configured in **Periodical Sending** [*1…255*][*s/min*] [*1…18*][*h*].

  - *Value Change*: the object will be sent every time the value changes.

  - *Target Value*: the object will be sent once the counter reaches the target value, configurable through the **Final Value** parameter [[*1…255*] / [*1…65535*]].

Join and send us your inquiries
about Zennio devices:

http://support.zennio.com

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

Tel. +34 925 232 002

www.zennio.com
info@zennio.com