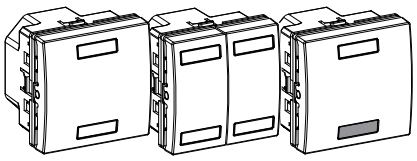
en

Schneider Electric

KNX push-button

Operating instructions



KNX push-button 1-gang
Art. no. ALB4x150

KNX push-button 2-gang
Art. no. ALB4x151

KNX 1-gang push-button with IR receiver
Art. no. ALB4x152

Accessories

– IR remote control Distance 2010
(Art. no. MTN570222)

For your safety

DANGER
Risk of fatal injury due to electrical current
All work on the device must only be carried out by trained and skilled electricians. Observe the country-specific regulations as well as the valid KNX guidelines.

Push-button introduction

Depending on the push-button, you have either two or four operating surfaces available to which you assign different functions via the ETS.

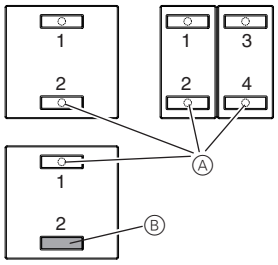
For example, you can:

- Switch and toggle
- Dimming
- Control blinds
- Save and retrieve scenes
- Call up linear regulator functions
- Save edge functions

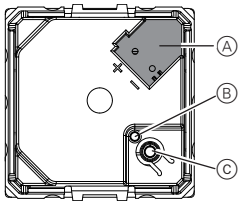
If required, you can disable the buttons and define the type of disabling.

The push-button with an IR receiver will allow you to operate each push-button by IR remote control as well.

Connections, displays and operating elements



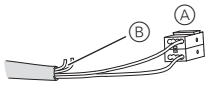
Ⓐ Status LEDs
Ⓑ IR receiver (no status LED)
1-4 Button assignment in the ETS



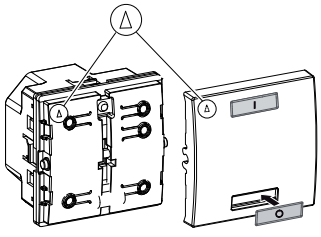
Ⓐ Bus connection
Ⓑ Programming LED
Ⓒ Programming button

Mounting the push-button

① Connect the red bus wire to the red terminal (+) and the black bus wire to the dark grey terminal (-) Ⓐ.



② Store the screen and the stability wire, as well as the white and yellow bus wire Ⓑ. They are not required.
③ Connect the terminal to the bus connection.
④ Fasten the push-button.
⑤ Put on the rockers.



⑥ Put on the frame.

Operating the push-button

① Make the desired settings in the ETS.
② Press the programming button.

The programming LED lights up.

③ Load the physical address and application into the device from the ETS.

The programming LED goes out.

Operating the push-button with a remote control

A push-button with an IR receiver will allow you to operate each push-button by IR remote control as well.

Assignment and operation:

Channel 1 = key 1 and IR remote control

Channel 2 = key 2 and IR remote control

Channels 3 to 9 = IR remote control

Teaching push-button to the Schneider remote control

The remote control and the push-button are set to each other. No learning procedure is necessary.

Teaching push-button to another remote control

① Press the upper key 10 times.

The status LED blinks first for 1 second, then it starts to flash.

Now you can teach channel 1:

② Press the remote control key 1 second long several times until the status LED lights up.

After 3 seconds, the status LED goes out and the channel is learned.

As soon as a channel has been learned, the push-button automatically switches to the next channel and the status LED starts to flash. Now you can teach channel 2.

Skipping a channel:

① Press the upper key 1 times.

The status LED lights up briefly; the channel was skipped. The status LED starts to flash again. Now you can teach the channel.

Ending the learning procedure:

- Press the upper key once.
- Automatically 30 s after the last push-button action
- Automatically after the last channel was learned

The learning mode was exited when the status LED blinks for 1 second.

Alternatively, you can also control the procedure via the "Activating - learning IR" object in the ETS.

Technical data

Power supply:	DC 24 V
KNX connection:	bus connecting terminal
Display elements:	Status LEDs
	1 programming LED
Operating elements:	Control keys
	1 programming button
Ambient operating temperature:	-5 °C to +45 °C
IR receiver	
Angle of reception:	approx. 60°
Reception range:	Dependent on the IR remote control used
IR channels:	9
Type of protection:	IP 20
Initialisation:	The device is ready for operation after 5 to 10 seconds.

Schneider Electric Industries SAS

If you have technical questions, please contact the Customer Care Center in your country.

www.schneider-electric.com

This product must be installed, connected and used in compliance with prevailing standards and/or installation regulations. As standards, specifications and designs develop from time to time, always ask for confirmation of the information given in this publication.

V6276-565-00 09/09

Settings in ETS

Selection in the product database

Manufacturer:	Schneider Electric Industries SAS	
Product family:	2.2	Push Button, 2-gang
Product type:	2.2.6	Altira
Range name:	Universal 1825/1.0	
Media type:	Twisted Pair	
Product name:	KNX Push-button, 2-gang	
Order number:	ALB45151, ALB46151	

The application can only be operated with ETS3.

Application overview

Application	Vers.	Functions
Universal 1825/1.0	1	**Push-button functions:**
		Send toggle commands – 1 bit, 1 byte
		Send switching commands – 1 bit, 1 byte
		Dimming
		Blind control
		Send edge commands – 1 bit, 2 bit (priority), 4 bit, 1 byte (distinction between short/long operation)
		Send edge commands – 2 byte (distinction between short/long operation)
		Parameterise 8-bit linear regulator
		Retrieve scenes
		Set the parameters for the disable function for push-buttons
		Set the parameters for scenes in the scene module

Application Universal 1825/1.0

Function overview

You can use this application to program the 2-gang push-button with status LEDs. All buttons can be assigned different functions independently. You can do the following:

- Switch and toggle
- Dimming
- Control blinds
- Save and retrieve scenes
- Select a slider function
- Save edge functions

The status LEDs can also be utilised independently of one another and in a wide variety of ways.

If required, you can disable the buttons and define the type of disabling. The integrated scene module saves up to eight scenes, and up to eight actuator groups can be assigned to each of those scenes.

Group addresses

Group addresses are managed dynamically. Maximum number of group addresses and assignments: 150.

Notes on this documentation

This application enables you to implement a multitude of functions. However, which functions are possible in each individual case depends on the KNX devices being controlled (e.g. dimming actuators, switch actuators etc.). The functions described here therefore show only the settings for this device.

Many parameters and their settings are dependent on the settings you have already made for other parameters. This means that some parameters will appear or disappear and the values available for selection will change according to settings you have already made. These dependencies have not been shown in the table for reasons of clarity. All settings are always shown.

Configurable times (staircase timer, ON delay, OFF delay, cyclic intervals etc.) are set via the base and factor parameters. The actual time is given by the multiplication of the two values.
Example:
Base = 1 second * factor = 3 gives 3 seconds.

The **bold** values in a table are the values set during factory configuration.

Push-button information

On the “Push-button info” tab you can see which push-button names in ETS correspond to which push-buttons on the device. The names assigned cannot be changed.

Push-button info	
Parameter	
Push-button 1 =	Upper left push-button
Push-button 2 =	Lower left push-button
Push-button 3 =	Upper right push-button
Push-button 4 =	Lower right push-button

Sending toggle commands – 1-bit, 1-byte

Each time the button is pressed, the 1-bit object type first inverts the object value and then transmits it to the bus, in other words making a “0” into a “1”. If the same button is pressed again, the “1” turns back into a “0”. The device is thus switched on and off alternately. This switching behaviour is called “toggling”.

For 1-byte object types, you can set two values, which are transmitted alternately after each press of the button.

An update or change to the object values is possible via the bus when another sensor switches the actuator (e.g. via a two-way circuit or a central command). To prevent “incorrect” toggling, the state of the actuator (“1” or “0”) must be tracked in the push-button. To do this, connect the group address of the second sensor to the switch/value object of the push-button.

Two objects (1 bit/1 byte) can also be transmitted in any combination when the push-button is pressed.

Push-button X	
Parameter	**Settings**
Select push-button function	Toggle

Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of the switch/value object. When the 1 byte object type is used, the LED lights up if value 1 is greater than zero.
- Display the status of the status feedback object.

Parameters

Parameter	Settings
Number of objects	**One**
	Two
Object A/Object B	**1 bit**
	1 byte in steps 0% - 100%
	1 byte continuous 0 - 255
Value 1	**100%**, 90%, 80%, ..., 0%, 25%, 75%
	255, 254, 253, ...0
Value 2	**0%**, 10%, 20%, ... 100%, 25%, 75%
	0, 1, 2, 3, ... 255
Trigger status LED	Switched on
	Switched off
	From switch/value object A
	From switch/value object B
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when switch/value object A not equal to 0
	Flashes when switch/value object B not equal to 0
	Flashes when switch/value object A equals 0
	Flashes when switch/value object B equals 0
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Switch object A/B	1 bit	Low	WCT	Transmit/ receive
Push-button X	Value object A/B	1 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Sending switching commands – 1 bit, 1 byte

When a push-button is pressed, the following values can be sent via the switch/value object

- An ON or OFF telegram
- 1 byte values in steps (0% - 100%)
- 1 byte values, infinitely adjustable (0 - 255)
- Two objects at the same time (1-bit, 1-byte) in any combination

Push-button X	
Parameter	**Settings**
Select push-button function	**Switching**

Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of the switch/value object. When the 1 byte object type is used, the LED lights up if value 1 is greater than zero.
- Display the status of the status feedback object.

Parameters

Parameter	Settings
Number of objects	**One**
	Two
Object A/Object B	**1 bit**
	1 byte in steps 0% - 100%
	1 byte continuous 0 - 255
Value	**ON telegram**
	OFF telegram
	100%, 90%, 80%, ..., 0%, 25%, 75%
	255, 254, 253, ...0
Trigger status LED	Switched on
	Switched off
	From switch/value object A
	From switch/value object B
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when switch/value object A not equal to 0
	Flashes when switch/value object B not equal to 0
	Flashes when switch/value object A equals 0
	Flashes when switch/value object B equals 0
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Switch object A/B	1 bit	Low	WCT	Transmit/ receive
Push-button X	Value object A/B	1 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Dimming

You can use the dimming function for the following:

- Dim brighter **and** darker using **one push-button** (single-button dimming)
- Either dim brighter **or** darker. You need a second push-button to dim in the other direction (two-button dimming).

Push-button X	
Parameter	**Settings**
Select push-button function	Dimming

Status indication

The status LED can:

- Display the status of the switch object
- Light up when pressed (for a long period), and go out when released
- Be on or off continuously
- Flash
- Display the status of the status feedback object

Parameter	**Settings**
Trigger status LED	Switched on
	Switched off
	From switch object
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when status feedback object not equal to 0
	Flashes when status feedback object equals 0
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Common parameters for single-button and two-button dimming

You can use the corresponding push-button to switch the light on or off (brief press) or dim it (longer press, the exact period can be parameterised). When switching takes place, an ON/OFF telegram is sent via the switch object. When dimming, dimming up or dimming down is carried out via the 4-bit dimming object; the parameters for the dimming steps can be set. You can also transmit the relevant dimming step cyclically for a period of time which can be set as required.

Parameter	**Settings**
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Dimming direction	Brighter
	Darker
	Brighter and darker
Send dimming levels cyclically	Yes
	No
Cycle time = basis * factor	
Basis	**0.1 s,** 1 s, 1 min
Factor (3-255)	3 - 255, **8**

Single-button dimming

You can dim both lighter **and** darker and also switch both on **and** off using a single push-button.

The current switching or dimming direction is always dependent on the previous action, i.e. if switched off, a brief push of the button will switch the light on and vice versa, and if the light has been dimmed up, prolonged operation of the push-button will dim the light down again. On release after prolonged actuation, a stop telegram will be transmitted via the 4-bit dimming object, thus terminating the dimming procedure in the dimming actuator.

An update or change to the object value is possible via the bus when another sensor switches or dims the actuator (e.g. via a two-way circuit or a central command). To prevent “incorrect” switching/dimming activity, the state of the actuator must be tracked in the push-button. To do this, connect the group address of the second sensor to the switch/dimming object of the push-button.

A single command is sufficient to cycle through the dimming range. This dimming procedure can be used for most applications. The other possible dimming steps (1/2 - 1/64 brighter or darker) dim brighter or darker by the selected step. For example, if the step is set to 1/4, you would need to push the button for a prolonged period four times in succession to dim from minimum to maximum brightness.

Parameter	**Setting**
Dimming direction	Brighter and darker
Step dimming (brighter)	**To max. brightness**
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
Step dimming (darker)	**To min. brightness**
	1/2 darker
	1/4 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker

Two-button dimming

You can dim either lighter **or** darker and switch either on **or** off with a single push-button. A second push-button for the opposite direction must be parameterised.

You can specify whether a stop telegram is to be transmitted when the push-button is released. If you have enabled the transmission of a stop telegram, a stop telegram will be transmitted via the 4-bit dimming object when the push-button is released after prolonged actuation, thus terminating the dimming procedure in the dimming actuator.

A single command is sufficient to cycle through the dimming range. This dimming procedure can be used for most applications. The other possible dimming steps (1/2 - 1/64 brighter or darker) dim brighter or darker by the selected step. For example, if the step is set to 1/4, you would need to push the button for a prolonged period four times in succession to dim from minimum to maximum brightness.

Parameter	Setting
Dimming direction	Brighter
	Darker
Step dimming (brighter)	**To max. brightness**
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
Step dimming (darker)	**To min. brightness**
	1/2 darker
	1/4 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker
Stop telegram after release	**Yes**
	No

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Switch object	1 bit	Low	WCT	Transmit/ receive
Push-button X	Dimming object	4 bit	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Blind control

You can use the blind control function to do the following:

- Raise the blinds/adjust the slats using a single push-button and lower the blinds/adjust the slats using a second push-button (two-button blind operation).
- Move the blind using an individual push-button and adjust the slats (single-button blind operation).
- Move the blind to a pre-specified position.
- Move the blind back and forth between two previously specified positions.

Push-button X	
Parameter	**Setting**
Select push-button function	Blind

Status indication

The status LED can:

- Flash
- Light up when pressed, and go out when released
- Be on or off continuously
- Display the status of the status feedback object

Parameter	Setting
Trigger status LED	Switched on
	Switched off
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	ON after long operation/release = OFF
	Flashes
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Two-button blind operation

You can either raise **or** lower the blind with a single push-button.

When the corresponding push-button is pressed for a short time, a stop/step telegram is transmitted; when the push-button is pressed for a longer period (the exact period can be parameterised), a movement telegram is transmitted. With this function, you must parameterise a second push-button with the corresponding settings for blind movement in the opposite direction. Both push-buttons must be given the same group addresses.

Parameter	Setting
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Direction of movement, blind	Up
	Down

Single-button blind operation

You can both raise **and** lower the blind with a single push-button.

The current direction of movement of the blind, or the direction of the slat adjustment, always depends on the previous action, i.e. when the blind has just been lowered, it will be raised the next time the push-button is activated for a long period (the exact period can be parameterised).

When a stop/step telegram has been transmitted to adjust the slats, a stop/step telegram for the same direction of movement can be generated by pressing the push-button again, as long as this subsequent push-button action is carried out within a set time period (which can be parameterised). If that time period has elapsed, the direction of rotation of the slats will change when the push-button is pressed briefly.

The push-button can receive telegrams via the stop/step and movement object, and can generate corresponding telegrams when the push-button is pressed, according to the values received. An update or change to the object values is possible via the bus when another sensor switches the actuator (e.g. via a two-way circuit or a central command). To prevent “incorrect” movement, the state of the actuator must be tracked in the push-button. To do this, connect the group address of the second sensor to the stop/step and movement object of the push-button.

Parameter	Setting
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Direction of movement, blind	Up and Down
Pause for slat – change of direction 100 ms * factor (5-50)	5 - 50, **10**

Moving the blind to a pre-specified position

If the blind actuator is capable of moving to specific position, you can use this function to specify one or two positions to which the blind can be moved using 1 byte position values with a push-button action. The position values can be set in steps between 0% and 100%, or infinitely from 0 - 255.

When moving to a position, the set value for the blind position and the slat position is transmitted using a short (or long) push-button action.

To address two positions, enter the required blind position and slat position for both. Position value 1 is transmitted with a short push-button action, while position value 2 is transmitted with a long push-button action. No movement or stop/step objects exist with these set parameters.

Parameter	Setting
Direction of movement, blind	With positional values
Select number of positionings	One position (short operation) Two positions (distinction between short/long operation)
Positional value 1 (short operation)	**In steps of 0% - 100%** Continuous 0 - 255
Position of blind	**100%**, 90%, 80%, ..., 0%, 25%, 75% **255**, 254, 253, ...0
Position of slats	**0%**, 10%, 20%, ... 100%, 25%, 75% **0**, 1, 2, 3, ... 255
Positional value 2 (long operation)	**In steps of 0% - 100%** Continuous 0 - 255
Position of blind	**100%**, 90%, 80%, ..., 0%, 25%, 75% **255**, 254, 253, ...0
Position of slats	**0%**, 10%, 20%, ... 100%, 25%, 75% **0**, 1, 2, 3, ... 255

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Stop/step object	1 bit	Low	WCT	Transmit/ receive
Push-button X	Movement object	1 bit	Low	WCT	Transmit/ receive
Push-button X	Blind position	1 byte	Low	CT	Transmit
Push-button X	Slat position	1 byte	Low	CT	Transmit
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Sending edge commands – 1 bit, 2 bit (priority), 4 bit, 1 byte

With this edge function you can transmit one or two objects simultaneously, and select the size of the objects required as needed (1 bit, 2 bit priority control, 4 bit or 1 byte in steps or infinitely). A distinction is made between the normal edge function and the extended edge function:

- With the normal edge function, you can specify which actions should be carried out when a push-button is pressed, and which should be carried out when a push-button is released.
- With the extended edge function you can also parameterise different actions to take place upon short and long operation of the push-button.

Push-button X	
Parameter	**Setting**
Select push-button function	Edges 1 bit, 2 bit (prio), 4 bit, 1 byte values
Select edge function	**Normal (operate, release)** Extended (+ long and short operation)

Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of object A/B.
- Display the status of the status feedback object.

Parameter	**Setting**
Trigger status LED	Switched on
	Switched off
	From object A
	From object B
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when object A not equal to 0
	Flashes when object B not equal to 0
	Flashes when object A equals 0
	Flashes when object B equals 0
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Normal edge function

With the normal edge function, you can specify which actions should be carried out when a push-button is pressed, and which should be carried out when a push-button is released. These actions could include:

- Send 1 or 0 (with 1 bit)
- Send value 1 or value 2 (with 2 bit, 4 bit or 1 byte): You can enter two values and set whether and how they are to be transmitted.
- Object sends its value:
  The object transmits the value which it currently has. Therefore you can, for example, transmit a value with the sending group address which was previously received via another group address.
- Toggle:
  The current object value is inverted and then transmitted. The device is thus switched on/off alternately or transmitted value 1/value 2 (toggling). The value can be modified via the bus.
- No action

The values available to you are 1 bit, 2 bit (priority control), 4 bit, 1 byte in steps or infinitely.

Push-button X	
Parameter	**Setting**
Edge function	Normal (operate, release)
Number of objects	**One**
	Two

Push-button X – edges object A/B	
Parameter	**Setting**
Object A/Object B	**1 bit**
	2 bit (priority control)
	4 bit
	1 byte in steps 0% - 100%
	1 byte continuous 0 - 255
Action on operation	**Sends 1**
	Sends 0
	Toggles
	Sends its value
	None
	Sends value 1
	Sends value 2
Action at release	Sends 1
	Sends 0
	Toggles
	Sends its value
	None
	Sends value 1
	Sends value 2

Push-button X – edges object A/B	
Parameter	**Setting**
Value 1	**Switch on with priority (11)**
	Switch off with priority (10)
	Remove priority control (00)
	Dim-darker-stop
	To min. brightness
	1/2 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker
	1/4 darker
	Dim-brighter-stop
	To max. brightness
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
	100%, 90%, 80%, ..., 0%, 25%, 75%
	255, 254, 253, ...0
Value 2	Switch on with priority (11)
	Switch off with priority (10)
	Remove priority control (00)
	Dim-darker-stop
	To min. brightness
	1/2 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker
	1/4 darker
	Dim-brighter-stop
	To max. brightness
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
	100%, 90%, 80%, ..., **0%**, 25%, 75%
	255, 254, 253, ...**0**

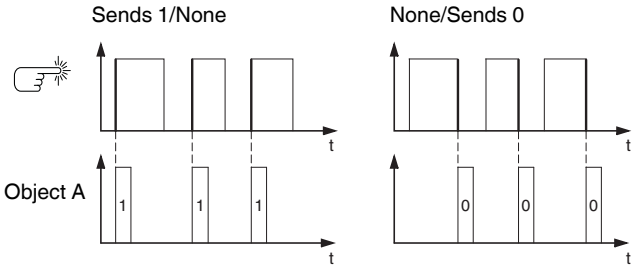
Principle of the edge function

Using the following diagrams, you can see how the edge function behaves when edges rise or fall.

The settings for “Action on operation/Action at release” are shown directly above each diagram.

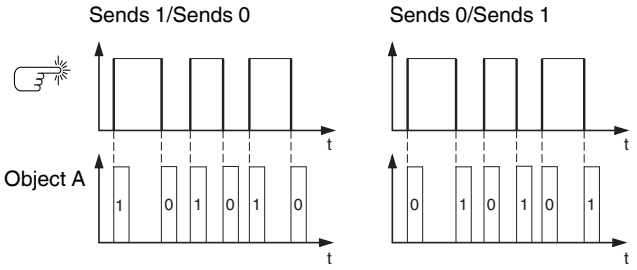
Example 1

Object A = 1 bit



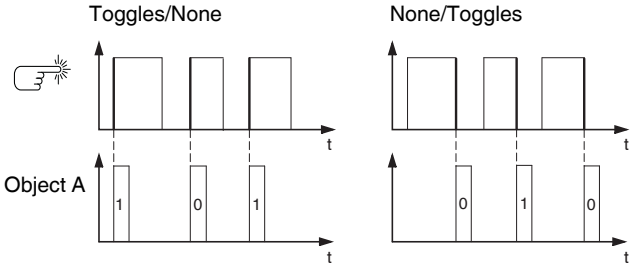
Example 2

Object A = 1 bit



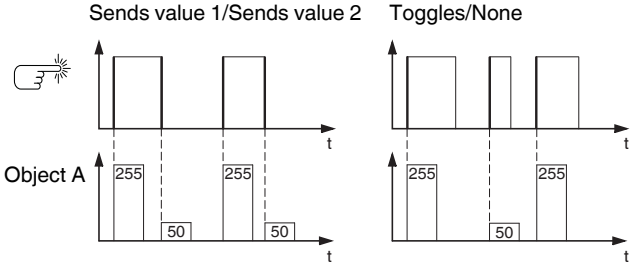
Example 3

Object A = 1 bit

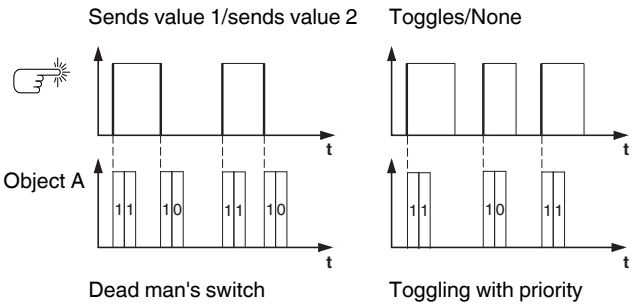


Example 4

Object A = 1 byte continuous 0 - 255
Value 1 = 255
Value 2 = 50



Example 5
Object A = 2 bit (priority control)
Value 1 = 11 (switch on with priority)
Value 2 = 10 (switch off with priority)



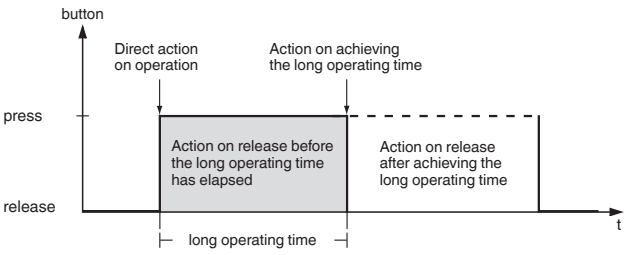
Extended edge function

With the extended edge function, you have a wider range of functions available. For example, you can set different actions for short and long presses of a push-button, both for when the push-button is pressed and for when it is released. You can also set a cycle time which can be parameterised for each object.

When parameterising, bear in mind that you need to set all four types of push-button operation (short/long press, pressing and releasing the button) in order to ensure that the push-button functions as required.

In order to read the object values, you may need to set the Read flags manually.

The following activation sequence chart shows the phases into which the pulse edge function is divided:



Push-button X	
Parameter	**Setting**
Edge function	Extended (+ long and short operation)
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Number of objects	**One** Two

Push-button X – edges object A/B	
Parameter	**Setting**
Object A/B	**1 bit** 2 bit (priority control) 4 bit 1 byte in steps 0% - 100% 1 byte contiuous 0 - 255
Direct action on operation	**Sends 1**
Action on release before the long operating time has elapsed	Sends 1 immediately and then cyclically
Action on achieving the long operating time	Sends 1 only cyclically
Action on release after achieving the long operating time	Sets object value to 1 (readable only) Sends 0 Sends 0 immediately and then cyclically Sends 0 only cyclically Sets object value to 0 (readable only) Sends value 1 Sends value 1 immediately and then cyclically Sends value 1 only cyclically Sets object value to value 1 (readable only) Sends value 2 Sends value 2 immediately and then cyclically Sends value 2 only cyclically Sets object value to value 2 (readable only) Toggles Toggles, sends immediately, then cyclically Toggles, only sends cyclically Toggles and is not sent Toggles cyclically, sends immediately, then cyclically Toggles cyclically, only sends cyclically Toggles cyclically and is not sent Sends its value Sends its value immediately and then cyclically Sends 1 and after a cycle time 0 Sends value 1, then value 2 after a cyclic time Cyclically increase the current object value by 1 Cyclically reduce the current object value by 2 None (stops cyclical sending) No change None (stop after current cycle time)

Push-button X – edges object A/B	
Parameter	**Setting**
Value 1	**Switch on with priority (11)**
	Switch off with priority (10)
	Remove priority control (00)
	Dim-darker-stop
	To min. brightness
	1/2 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker
	1/4 darker
	Dim-brighter-stop
	To max. brightness
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
	100%, 90%, 80%, ..., 0%, 25%, 75%
	255, 254, 253, ...0
Value 2	Switch on with priority (11)
	Switch off with priority (10)
	Remove priority control (00)
	Dim-darker-stop
	To min. brightness
	1/2 darker
	1/8 darker
	1/16 darker
	1/32 darker
	1/64 darker
	1/4 darker
	Dim-brighter-stop
	To max. brightness
	1/2 brighter
	1/4 brighter
	1/8 brighter
	1/16 brighter
	1/32 brighter
	1/64 brighter
	100%, 90%, 80%, ..., **0%**, 25%, 75%,
	255, 254, 253, ...**0**
Cycle time = basis * factor	
Basis	0.1 s, 1 s, **1 min**, 1 h, 1 day
Factor (3-255)	3-255, **10**

A description of the most important actions is given below:

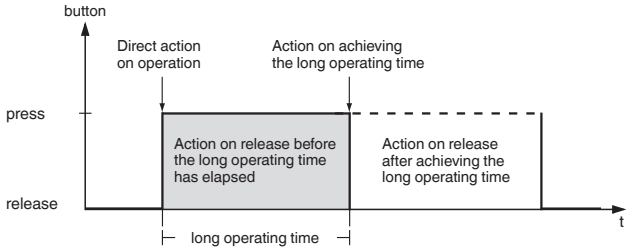
- Sends [value]:
  Transmits the current value and stops a cyclical transmission.
- Sends [value] immediately and then cyclically:
  If no cycle time is running, [value] is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is interrupted, [value] is transmitted and a new cycle time is started.
- Sends [value] only cyclically:
  If no cycle time is running, [value] is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; [value] is transmitted after the current cycle time has elapsed, and a new cycle time is started.
- Sets object value to [value] (readable only)
  [value] is written into the object and is not transmitted. Any active cycle time is terminated.
- Toggles:
  Compares the current object value with [value]. If both are the same, value 1 or value 2 is transmitted. If they are different, [value] is transmitted.
- Toggles, sends immediately, then cyclically:
  If no cycle time is running, the value is toggled (see “toggles”), transmitted immediately, and a new cycle time is started. If a cycle time is already running, it is interrupted, the toggled value is transmitted and a new cycle time is started. Subsequently, the value which has already been toggled is always transmitted cyclically.
- Toggles, only sends cyclically:
  If no cycle time is running, the toggled value is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; the toggled value is transmitted after the current cycle time has elapsed, and a new cycle time is started. Subsequently, the value which has already been toggled is always transmitted cyclically.
- Toggles and is not sent:
  The toggled value is written into the object and is not transmitted. Any active cycle time is terminated.
- Toggles cyclically, sends immediately, then cyclically:
  If no cycle time is running, the value is toggled (see “toggles”), transmitted immediately, and a new cycle time is started. If a cycle time is already running, it is interrupted, the toggled value is transmitted and a new cycle time is started. Subsequently, it is always toggled cyclically and the new value is transmitted.
- Toggles cyclically, only sends cyclically:
  If no cycle time is running, the toggled value is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; the toggled value is transmitted after the current cycle time has elapsed, and a new cycle time is started.

Subsequently, it is always toggled cyclically and the new value is transmitted.

- Toggles cyclically and is not sent:
  The toggled value is written into the object and is **not** transmitted. Subsequently, it is always toggled cyclically and the new value is written into the object.
- Sends its value:
  The current object value is transmitted. Any active cycle time is terminated.
- Sends its value immediately and then cyclically:
  If no cycle time is running, the current object value is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is interrupted, the current object value is transmitted and a new cycle time is started. Subsequently, the current object value is always transmitted cyclically.
- Cyclically increase the current object value by [value]:
  If no cycle time is running, [value] is added to the current object value, the object value is transmitted, and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; the current object value with [value] added is transmitted and a new cycle time is started.
- Reduce the current object value by [value] cyclically:
  If no cycle time is running, [value] is subtracted from the current object value, the object value is transmitted, and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; the current object value with [value] subtracted is transmitted and a new cycle time is started.
- Sends [value A] and after a cycle time [value B]:
  [value A] is transmitted immediately, and [value B] is transmitted after **one** cycle time, regardless of whether a cycle time is already running or not (staircase lighting timer function).
- None (stops cyclical sending):
  No action is carried out, and any active cycle time is stopped.
- No change:
  The current action remains unchanged (e.g. “sends value 1, then value 2 after a cycle time”).
- None (stop after current cycle time):
  No action is currently carried out, but any active cycle time is **not** stopped. It runs through until the end, and then transmits the corresponding value.

Examples of use for the edge function

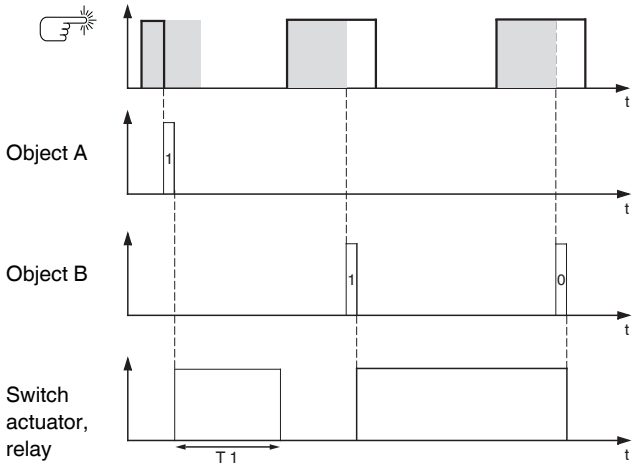
The following activation sequence chart shows the phases into which the pulse edge function is divided:



Staircase lighting function with cleaning light function

With a brief press of a push-button, the switch actuator switches on the light. A long press of the push-button extends the staircase lighting function (= cleaning light function) until a second long press of the button switches off the actuator. The switch actuator requires a staircase lighting function and a disable function for this function.

Number of objects = 2 (object A/B)
Object A/B = 1 bit
Object A: Action on release before the long operating time has elapsed = Sends 1
Object B: Action on achieving the long operating time = Toggles
Connect object A with the switch object and object B with the disable object of the switch actuator.



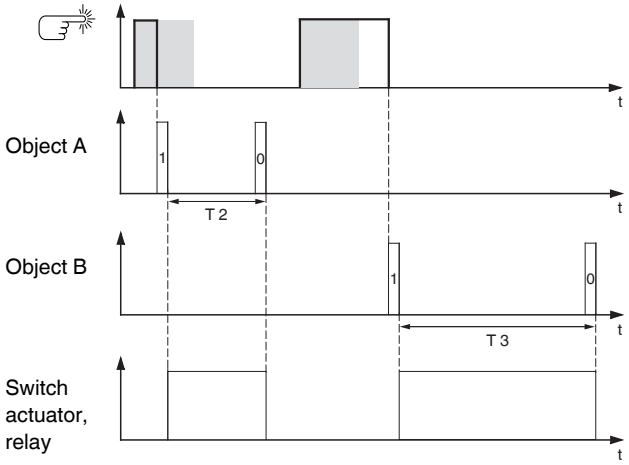
T 1 = Staircase timer period

Short and long staircase timer

You can use this function to produce a brief and a long staircase lighting time with the push-button. The switch actuator requires no staircase lighting function for this request.

With a brief press of the push-button, the switch actuator switches on the light, and after a parameterised cycle time (e.g. 3 minutes), it switches it back off again. With a long press of the push-button, the same function is carried out, but with a longer cycle time (e.g. 6 minutes).

Number of objects = 2 (object A/B)
Object A/B = 1 bit
Object A: Action on release before the long operating time has elapsed = Sends 1 and after a cycle time 0.
Cycle time = e.g. 3 minutes
Object B: Action on release after achieving the long operating time = Sends 1 and after a cycle time 0.
Cycle time = e.g. 6 minutes
Connect object A and object B with the switch object of the switch actuator.



T 2 = Short cycle time
T 3 = Long cycle time

Switching the light on/off permanently, or switching off after a cycle time has elapsed

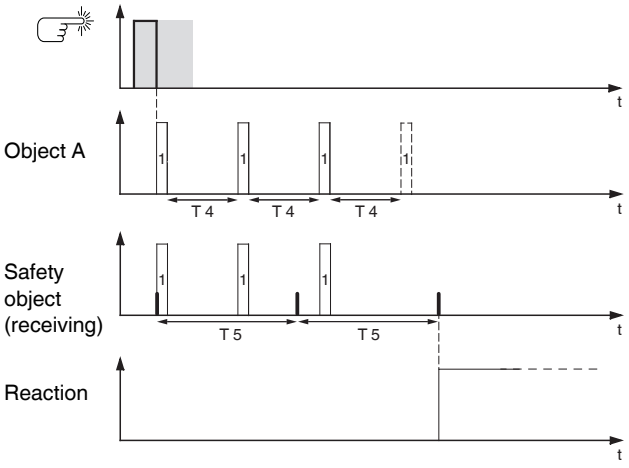
With a brief press of a push-button, the switch actuator switches the light on or off permanently. With a long press of a push-button, the light switches on, and after a parameterised cycle time (e.g. 6 minutes), it switches back off again. Due to the cycle time in the push-button which can be parameterised, the switch actuator requires no staircase lighting function for this function.

Number of objects = 2 (object A/B)
Object A/B = 1 bit
Object A: Action on release before the long operating time has elapsed = toggles
Object B: Action on achieving the long operating interval = sends 1 and after a cycle time 0.
Action on release after achieving the long operating time = no change.
Cycle time = e.g. 6 minutes.
Connect object A and object B with the switch object of the switch actuator.

Electronic protection against theft

This example will show you how to program electronic theft protection for the push-button. It is activated by a brief push-button action and then transmits cyclically. As soon as the push-button is forcibly separated from the bus, this can be reported or an alarm can be triggered.

Number of objects = 1 (object A)
Object A = 1 bit
Object A: Action on release before the long operating time has elapsed = Sends 1 immediately and then cyclically.
Action on achieving the long operating time = No change.
Action on release after achieving the long operating time = No change.
Cycle time = e.g. 10 minutes.
Connect object A with an object that listens cyclically for telegrams (e.g. a safety object). The monitoring time set on the safety object must be longer than the cycle time of the push-button. If the safety object receives no telegrams from the push-button during this time, a reaction which can be parameterised is activated (e.g. channel is switched on).

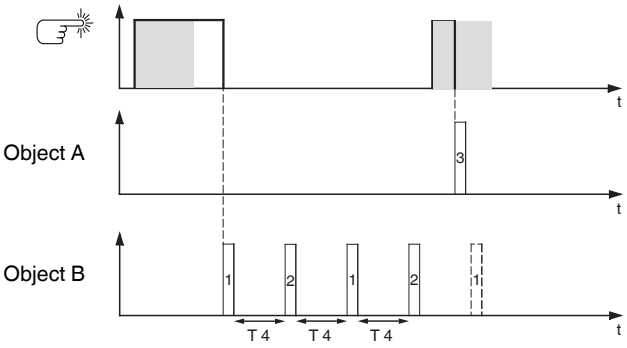


T 4 = Cycle time
T 5 = Monitoring time

Effect lighting

This example shows you how to program effect lighting, for example for a display window. A long push-button action switches between two different lighting scenes. A short push-button action stops the toggling and transmits a scene which switches off everything. The scene module of the push-button is used to retrieve the scene.

Number of objects = 2 (object A/B)
Object A/B = 1 byte contiuous 0 - 255
Object A: Direct action on operation = None (stops cyclical sending).
Action on release before the long operating time has elapsed = Sends value 1.
Action on achieving the long operating time = None (stops cyclical sending).
Action on release after achieving the long operating time = None (stops cyclical sending).
Value 1 = 3
Object B: Direct action on operation = None (stops cyclical sending).
Action on release before the long operating time has elapsed = None (stops cyclical sending).
Action on achieving the long operating time = None (stops cyclical sending).
Action on release after achieving the long operating time = Toggles cyclically, sends immediately, then cyclically.
Value 1 = 1
Value 2 = 2
Cycle time = e.g. 1 minute.
Connect object A and object B with the extension unit object of the scene function.



T 4 = Cycle time

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Object A/B	1 bit	Low	WCT	Transmit/ receive
Push-button X	Object A/B	2 bit	Low	WCT	Transmit/ receive
Push-button X	Object A/B	1 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Sending edge commands – 2 byte

With this edge function, you can send a 2-byte object in floating point format or in integer format (with or without sign). A distinction is made between the normal edge function and the extended edge function:

- With the normal edge function, you can specify which actions should be carried out when a push-button is pressed, and which should be carried out when a push-button is released.
- With the extended edge function, you can also set the actions before and after the long button actuation period is completed.

Push-button X	
Parameter	**Setting**
Select push-button function	Edges with 2 byte values
Select edge function	**Normal (operate, release)**
	Extended (+ long and short operation)

Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of the status feedback object.

Parameter	Setting
Trigger status LED	Switched on
	Switched off
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Normal edge function

With the normal edge function, you can specify which actions should be carried out when a push-button is pressed, and which should be carried out when a push-button is released. These actions could include:

- Send value 1 or value 2:
  You can specify two values and set whether and how they are to be transmitted.
- Object sends its value:
  The object transmits the value which it currently has. Therefore you can, for example, transmit a value with the sending group address which was previously received via another group address.
- No action

Available values are the floating point value or integer values with/without sign.

Push-button X	
Parameter	**Setting**
Select edge function	**Normal (operate, release)**
Action on operation	**Sends value 1**
	Sends value 2
	Sends its value
	None
Action at release	Sends value 1
	Sends value 2
	Sends its value
	None

Push-button X – edges values	
Parameter	**Setting**
Object type value	**Floating point**
	Integer with sign (-32768...32767)
	Integer without sign (0 … 65535)
Value 1 = basis * factor	
Basis (possible values in brackets)	0,01, ... 327,68; **0,01**
Factor (0-2047)	0 - 2047, **1000**
Value 2 = basis * factor	
Basis (possible values in brackets)	0,01, ... 327,68; **0,01**
Factor (0-2047)	0 - 2047, **2000**
Value 1 (-32768 - 32767)	-32768...32767, **32767**
Value 2 (-32768 - 32767)	-32768...32767, **-32768**
Value 1 (0-65535)	0-65535, **65535**
Value 2 (0-65535)	0-65535, **0**

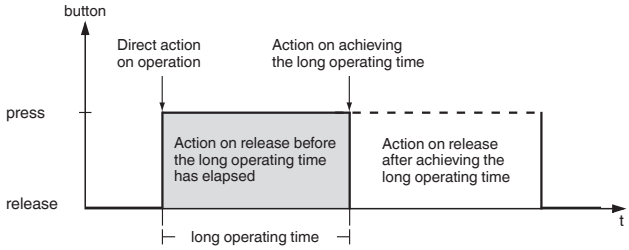
Extended edge function

With the extended edge function, you have a wider range of functions available. For example, you can set different actions for short and long presses of a push-button, both for when the push-button is pressed and for when it is released. You can also set a cycle time which can be parameterised for the object.

When parameterising, bear in mind that you need to set all four types of push-button operation (short/long press, pressing and releasing the button) in order to ensure that the push-button functions as required.

In order to read the object values, you may need to set the Read flags manually.

The following activation sequence chart shows the phases into which the pulse edge function is divided:



Push-button X	
Parameter	**Setting**
Select edge function	Extended (+ long and short operation)
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Direct action on operation	**Sends value 1**
Action on release before the long operating time has elapsed	Sends value 1 immediately and then cyclically
Action on achieving the long operating time	Sends value 1 only cyclically
Action on release after achieving the long operating time	Sets object value to value 1 (readable only)
	Sends value 2
	Sends value 2 immediately and then cyclically
	Sends value 2 only cyclically
	Sets object value to value 2 (readable only)
	Sends its value
	Sends value 1, then value 2 after cycle time
	None (stops cyclical sending)
	No change
Cycle time = basis * factor	
Basis	0.1 s, 1 s, **1 min**, 1 h, 1 day
Factor (3-255)	3-255, **10**

A description of the actions is given below:

- Sends [value]:
  Transmits the current value and stops a cyclical transmission.
- Sends [value] immediately and then cyclically:
  If no cycle time is running, [value] is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is interrupted, [value] is transmitted and a new cycle time is started.
- Sends [value] only cyclically:
  If no cycle time is running, [value] is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is **not** interrupted; [value] is transmitted after the current cycle time has elapsed, and a new cycle time is started.
- Sets object value to [value] (readable only)
  [value] is written into the object and is not transmitted. Any active cycle time is terminated.
- Sends its value:
  The current object value is transmitted. Any active cycle time is terminated.
- Sends [value A] and after cycle time [value B]:
  [value A] is transmitted immediately, and [value B] is transmitted after **one** cycle time, regardless of whether a cycle time is already running or not (staircase lighting timer function).
- None (stops cyclical sending):
  No action is carried out, and any active cycle time is stopped.
- No change:
  The current action remains unchanged (e.g. “sends value 1, then value 2 after a cycle time”).

Push-button X - edges, values	
Parameter	**Setting**
Object type value	**Floating point** Integer with sign (-32768...32767) Integer without sign (0 … 65535)
Value 1 = basis * factor	
Basis (possible values in brackets)	0,01, ... 327,68; **0,01**
Factor (0-2047)	0 - 2047, **1000**
Value 2 = basis * factor	
Basis (possible values in brackets)	0,01, ... 327,68; **0,01**
Factor (0-2047)	0 - 2047, **2000**
Value 1 (-32768 - 32767)	-32768...32767, **32767**
Value 2 (-32768 - 32767)	-32768...32767, **-32768**
Value 1 (0-65535)	0-65535, **65535**
Value 2 (0-65535)	0-65535, **0**

Communication objects

You can select the following communication objects:

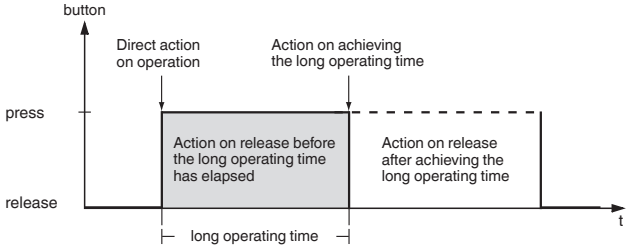
Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Value object A	2 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Setting the parameters for the 8 bit slider

With this function you can program a push-button as a slider, allowing you to automatically increase or reduce object values cyclically (for example). The slider function can be parameterised with or without limit values for all four actions: when pressing/releasing and with a short or long button operating time (brief/long press).

Push-button X	
Parameter	**Setting**
Select push-button function	8 bit slider
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**

The following activation sequence chart shows the phases into which the slider function is divided:



Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of the status feedback object.
- Display the status of the value object.

Parameter	**Setting**
Trigger status LED	Switched on
	Switched off
	From value object A
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when value object A not equal to 0
	Flashes when value object A equals 0
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF

Push-button X slider	
Parameter	**Setting**
Slider function	**With limit values**
	Without limit values
Direct action on operation	Send value 1, then increase cyclically by step width
Action on release before the long operating time has elapsed	Send value 2, then reduce cyclically by step width
Action on achieving the long operating time	Increase current object value cyclically
Action on release after achieving the long operating time	Increase current object value once
	Reduce current object value cyclically
	Reduce current object value once
	Reverse slide direction and send cyclically
	Reverse slide direction and increase/decrease cyclically
	Stepwise to the limit values and back again
	Increase stepwise within limits
	Decrease stepwise within limits
	None (stops cyclical sending)
	no change
Value 1	0 - 255, **0**
Set step value	0 - 255, **10**
Value 2	0 - 255, **100**
Cycle time = basis * factor	
Basis	**0.1 s**, 1 s, 1 min, 1 h, 1 day
Factor (3-255)	3-255, **5**

A description of the actions is given below:

- Send value 1, then increase cyclically by step width:
  If no cycle time is running, value 1 is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is interrupted, value 1 is transmitted and a new cycle time is started.
- Send value 2, then reduce cyclically by step width:
  If no cycle time is running, value 2 is transmitted immediately and a new cycle time is started. If a cycle time is already running, it is interrupted, value 2 is transmitted and a new cycle time is started.
- Increase current object value cyclically:
  Increase the current object value cyclically by the parameterised step value.
- Increase current object value once:
  Increase the current object value once by the parameterised step value. Any active cycle time is terminated.
- Reduce current object value cyclically:
  Reduce the current object value cyclically by the parameterised step value.
- Reduce current object value once:
  Reduce the current object value once by the parameterised step value. Any active cycle time is terminated.

- Reverse slide direction and send cyclically:
  If no cycle time is running, the slide is pushed in the opposite direction (of this push-button) and a new cycle time is started. If a cycle time is already running, it is interrupted, the slide is immediately pushed in the opposite direction (of this push-button) and a new cycle time is started. Cyclic transmission is stopped when the maximum/minimum value is reached.
- Reverse slide direction and increase/decrease cyclically:
  If no cycle time is running, the slide is pushed in the opposite direction (of this push-button) and a new cycle time is started. If a cycle time is already running, it is interrupted, the slide is immediately pushed in the opposite direction (of this push-button) and a new cycle time is started. Cyclic transmission is not stopped when the maximum/minimum value is reached. When an incrementing value reaches the maximum value, the value is set to the minimum value and cyclic transmission continues. When an decrementing value reaches the minimum value, the value is set to the maximum value and cyclic transmission continues.
- Stepwise to the limit values and back again:
  The limit values are approached by one step at a time. When a limit value is reached, the sliding direction is reversed for the next action.
- Increase stepwise within limits:
  The value is incremented by one step value at a time, within the limits. The limits are not exceeded; instead value 1 is sent again after the last possible step.
  Example: Value 1: “0”, value 2: “255”, step size: “100”; the following values are sent: 39%, 78%, 0%, 39%, 78%, 0%, etc.
- Decrease stepwise within limits:
  The value is reduced by one step value at a time, within the limits. The limits are not exceeded; instead value 2 is sent again after the last possible step.
  Example: Value 1: “0”, value 2: “255”, step size: “100”. The following values are sent: 100%, 61%, 22%, 100%, 61%, 22%, etc.
- None (stops cyclical sending):
  No action is carried out, and any active cycle time is stopped.
- No change:
  No action is carried out, and any active cycle time is continued.

Keeping within the limits and toggling to a new slide direction are only possible with local, on-site operation!

Example: Implementing a step dimmer with slider function

It is possible to dim a dimming actuator in several “steps” using a push-button. Push-button 1 is used as an 8 bit slider. The status LED can be controlled by the status feedback object of the dimmer.

“Push-button 1” tab:

Push-button function = 8 bit slider

“Push-button 1 slider” tab:

Slider function: “With limit values”

Direct action on rocker operation = Stepwise to the limit values and back again

Action on release, on or after achieving the long operating time = No change

Value 1 = 0

Step value = 51

Value 2 = 255

The cycle time is not required for this function.

Connect the push-button value object to the dimming actuator value object.

Every new press of the push-button sends a new dimming value, in the following steps: 20%, 40%, 60%, 80%, 100%, 80%, 60%, 40%, 20%, 0%, 20%, etc.

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Value object A	1 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Retrieving scenes

Retrieving scenes by push-button does not access the internal scene module, but rather only accesses the bus externally via communication objects. If you therefore wish to retrieve scenes stored in the internal scene module using a push-button, you must connect the corresponding communication object with the extension unit object of the scene function.

There are two types of scene function:

- Normal
- Extended

Push-button X	
Parameter	**Setting**
Select push-button function	Scene
Select scene function	**Normal (short = recall/ long = save)** Extended

Status indication

The status LED can:

- Be switched on or off continuously.
- Light up when pressed (for a long period), and go out when released.
- Flash.
- Display the status of the status feedback object.
- Display the status of object A/B.

Parameter	**Setting**
Trigger status LED	Switched on
	Switched off
	From status feedback object
	Operation = ON/release = OFF
	Long operation = ON/release = OFF
	Flashes
	Flashes when status feedback object equals 1
	Flashes when status feedback object equals 0
	Operation = flash/release = OFF
	Long operation = flash/release = OFF
	From object A
	From object B
	Flashes when object A not equal to 0
	Flashes when object B not equal to 0

Normal scene function

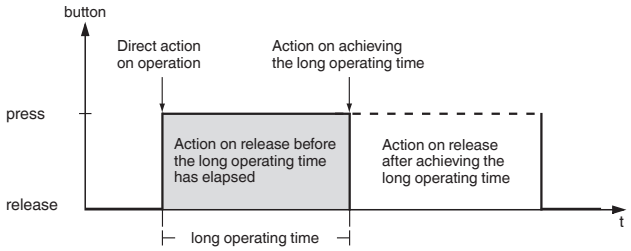
With the normal scene function, a scene is retrieved by a brief push-button action and a long push-button action is used to save a scene. You merely have to set the time after which a push-button action is identified as being long, together with the triggering of the status LED and the scene address.

Push-button X	
Parameter	**Setting**
Select scene function	Normal (short = recall/ long = save)
Long operation time equals 100 ms * factor (4 - 250)	4 - 250, **6**
Scene address (0-63)	0-63, **0**

Extended scene function

With the extended scene function, you can set different actions for short and long presses of a push-button, both for when the push-button is pressed and for when it is released. You can also set a cycle time which can be parameterised for each object.

The following activation sequence chart shows the phases into which the scene function is divided:



Push-button X	
Parameter	**Setting**
Select scene function	Extended
Long operation defined as 100 ms * factor (4 - 250)	4 - 250, **30**
Number of objects	**one** two

Push-button X – scene object A/B	
Parameter	**Setting**
Direct action on operation	Sends value 1
Action on release before the long operating time has elapsed	Sends value 2
Action on achieving the long operating time	Toggles
Action on release after achieving the long operating time	Toggles cyclically, sends immediately, then cyclically
	Sends value 1, then value 2 after a cycle time
	None (stops cyclical sending)
	No change
Value 1 Scene address (0-63)	0-63, **0**
Value 1 to retrieve/save the scene	**Retrieve**
	Save
Value 2 Scene address (0-63)	0-63, **0**
Value 2 to retrieve/save the scene	Retrieve
	Save
Cycle time = basis * factor	
Basis	0.1 s, **1 s**, 1 min, 1 h, 1 day
Factor (3-255)	3-255, **10**

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Push-button X	Object A	1 byte	Low	WCT	Transmit/ receive
Push-button X	Object B	1 byte	Low	WCT	Transmit/ receive
Push-button X	Status feedback object	1 bit	Low	WC	Receive

Setting the parameters for the disable function for push-buttons

You can use the disable function to disable the push-buttons in three different ways:

1. For each push-button separately
2. All push-buttons function like a predefined master push-button
3. Toggle between two local scenes.

You can determine whether disabling should occur when disable object = 0 or when disable object = 1.

When a disable function is activated via the disable object, all current push-button functions (including cyclical actions) are suppressed.

Disable function for push-buttons	
Parameter	**Setting**
Apply disable function	**No**
	Yes
Set disable function	
Execute disable function	At object value 0
	At object value 1
Type of blocking	Set separately for each push-button
	All push-buttons function like master
	Toggle between two scenes (scene addresses)

For each push-button separately

With this function you can disable each push-button individually. When a push-button is disabled, it does not execute a function when pressed.

Disable function for push-buttons	
Parameter	**Setting**
Type of blocking	Set separately for each push-button
Push-button 1 disable Push-button 2 disable Push-button 3 disable Push-button 4 disable	**Yes** No

All push-buttons function like master

You can use this function to specify one push-button as a master push-button. When any push-button is pressed, the function that was parameterised for the master key is carried out.

Disable function for push-buttons	
Parameter	**Setting**
Type of blocking	All push-buttons function like master
Master push-button =	Push-button 1 Push-button 2 Push-button 3 Push-button 4

Toggle between two scenes (scene addresses)

With this action you can toggle between two scenes that are parameterised in the scene module. When any push-button is pressed, one or the other scene is retrieved in alternation.

The scene addresses entered must be known to the push-button's internal scene module, and must be identical to the scene addresses in the module. The scene addresses entered with this function are not transmitted to the bus.

Disable function for push-buttons	
Parameter	**Setting**
Type of blocking	Toggle between two scenes (scene addresses)
First scene address	0-63, **0**
Second scene address	0-63, **1**

Communication objects

You can select the following communication objects:

Function	**Object name**	**Type**	**Prio**	**Flags**	**Behaviour**
Disable function	Locking object	1 bit	Low	WC	Receive

Setting the parameters for scenes in the scene module

The push-button is fitted with its own scene module, which enables you to save up to eight scenes permanently. The saved scenes can be overwritten if you have parameterised a release for this purpose.

The entire scene function is controlled via the extension object (1 byte). The following objects are also available for sending scene values to the bus:

- An object for programming release
- Eight objects for values with 1 bit, 2 bit and 1 bytes
- One object (Actuator group 7) for values with 2 bytes

You can set the time between the actuator read telegrams. This makes sense, e.g. when the anticipated response can last a long time (line coupler, area coupler).

If a read request is lost, or is not responded to, the current object value is saved in the scene (either through a read request, or written via an output). To check the correct saving procedure, you should retrieve the scene last saved on the push-button. If this remains unchanged, the individual saving procedure has been completed free of errors. If there is a difference, this means that a read request was not responded to correctly.

If the push-button works through a scene, and a further scene is retrieved, the current process is interrupted and the last retrieved scene is worked through.

Scene module	
Parameter	**Setting**
Apply scene module	**No** Yes
Save scenes	**Yes** Yes, if enable object = 1 No
Time between 2 read telegrams 100 ms * factor (2-255)	2-255, **10**

Specifying scene actuator groups

In this card, you can specify the data type of the eight actuator groups. Actuator group 7 is a special group which allows you to transmit values with 16 bits.

Scene actuator groups	
Parameter	**Setting**
Object types of the actuator groups	
Actuator group 1 Actuator group 2 Actuator group 3 Actuator group 4 Actuator group 5 Actuator group 6 Actuator group 8	**Switch object** Value object (8 bit in steps) Value object (8 bit stepless) Priority object
Actuator group 7 (also 16 bit possible)	**Switch object** Value object (8 bit in steps) Value object (8 bit stepless) Priority object Value object (16 bit without sign) Value object (16 bit with sign) Value object (16-bit floating point value)

Specifying scene addresses and values

For each scene, you specify the scene address via which the scene on the extension object should be retrieved. You also specify the time between the individual scene telegrams.

Make sure that you always enter unique scene addresses for this device, i.e. no scene address should be allocated more than once.

Scene X	
Parameter	**Setting**
Scene address (0-63)	0-63
Time between scene telegrams 100 ms * factor (2-255)	2-255, **10**

Finally, specify the actuator groups and their values for this scene. These only remain valid up to the first time the scene is saved.

The value range which can be set depends on the data type set for the “scene actuator groups”.

Scene X - values	
Parameter	**Setting**
Value 1 sending Value 2 sending Value 3 sending Value 4 sending Value 5 sending Value 6 sending Value 8 sending	ON telegram OFF telegram **No telegram** 0% - 100% 0 - 254 Switch on with priority (11) Switch off with priority (10) Remove priority (00)
Value 7 sending	ON telegram OFF telegram **No telegram** 0% - 100% 0 - 254 Switch on with priority (11) Switch off with priority (10) Remove priority (00) Send telegram
Value 7 sending (0-65535)	0-65535, **65535**
Value 7 sending (-32768-32767)	-32768...32767, **32767**
Value 7 = basis * factor Basis (possible values in brackets) Factor (0-2047)	 0,01...327,68, **0,01** 0-2047, **1000**

Communication objects

You can select the following communication objects:

Function	Object name	Type	Prio	Flags	Behaviour
Save scenes	Enable object	1 bit	Low	WC	Receive
Scene function	Extension object	1 byte	Low	WC	Receive
Switching	Actuator group 1-8	1 bit	Low	WCT	Transmit/ receive
Transmit value	Actuator group 1-8	1 byte	Low	WCT	Transmit/ receive
Transmit value	Actuator group 7	2 byte	Low	WCT	Transmit/ receive
Priority operation	Actuator group 1-8	2 bit	Low	WCT	Transmit/ receive

Behaviour on bus voltage recovery/ bus voltage failure

Behaviour on bus voltage recovery

Depending on the setting the status LEDs may be switched on, switched off or may flash.

Telegrams are not sent.

Behaviour on bus voltage failure

The LEDs will be switched off.

Schneider Electric Industries SAS

If you have technical questions, please contact the Customer Care Center in your country.

www.schneider-electric.com

This product must be installed, connected and used in compliance with prevailing standards and/or installation regulations. As standards, specifications and designs develop from time to time, always ask for confirmation of the information given in this publication.