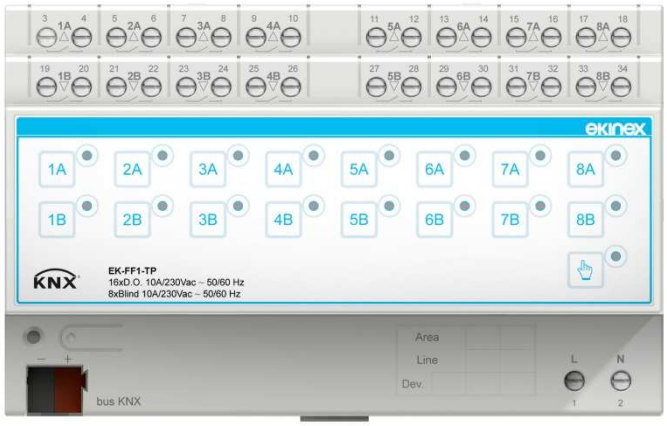
ekinex

CONTROL YOUR LIVING SPACE

Application manual



KNX binary output module

EK-FE1-TP 4/8-channel

EK-FF1-TP 8/16-channel

Contents

1 Scope of the document

This application manual describes application details for the A1.0 release of the ekinex® KNX binary output modules EK-FE1-TP (4/8 channels) and EK-FF1-TP (8/16 channels).

The document is aimed at the system configurator as a description and reference of device features and application programming. For installation, mechanical and electrical details of the device please refer to the technical description datasheet.

Application manual and application programs for ETS are available for download at www.ekinex.com.

Item	*File name (## = release)*	*Version*	*Device rel.*	*Update*
Technical datasheet	STEKFE1TP_EN.pdf STEKFF1TP_EN.pdf	-	A1.0	02 / 2014
Application manual	MAEKFE1FF1TP_EN.pdf	-		
Application program	APEKFE1TP##.knxprod APEKFF1TP##.knxprod	-		

You can access the most up-to-date version of the full documentation for the device using following QR codes:

For the 4-channel interface EK-FE1-TP:

For the 8-channel interface EK-FF1-TP:

2 Product description

The ekinex® binary output modules EK-FE1-TP and EK-FF1-TP are S-mode KNX modular devices for independent switching respectively of 8 or 16 electrical loads; to this purpose, the outputs of the devices are equipped with potential-free relay contacts.

The two devices differ only for the number of the output channels; their operation is the same in every respect, except for the fact that, for the smaller unit, the parameters and communication objects bound to the upper 8 channels are not available.

In this manual, for the sake of clarity, the larger 8/16 channel unit is referenced; only where differences between the two types of units exist, they will be explicitly highlighted.

The device is equipped with an integrated bus communication module and is designed for rail mounting in distribution boards.

For operation, the devices receives KNX telegrams from the bus, sent by another KNX device (such as a pushbutton, a sensor, a display, a timer, etc.); these telegrams cause the activation or deactivation of one or more relays.

Manual operation of an output channel is also possible by using the corresponding membrane keys on the front side; indicator LEDs display the switching status of the relays.

The status of the outputs is maintained even in case of failure of the bus voltage, provided that the auxiliary supply voltage does not also fail, and unless the device is programmed otherwise..

The device is powered by the KNX bus line with a 30 V DC SELV voltage only as far as the bus interface is concerned; for all other internal operation voltages, a 230 V AC power supply is required which in most cases can be easily derived from the wiring already in place for power loads.

For further technical information, please also refer to the product datasheets STEKFE1TP_EN.pdf and STEKFF1TP_EN.pdf available on the ekinex website www.ekinex.com.

3 Switching, display and connection elements

The device is equipped with:

- Membrane keys for manual operation
- A membrane key to switch between manual and online mode
- LED indicators for the status of the outputs and for the indication of manual mode
- A programming pushbutton and a programming LED
- Terminals for the connection of output loads
- Terminals for the KNX bus line connection
- Terminals for the connection of the auxiliary power supply

The terminals of outputs that can coupled in a pair are placed on corresponding positions in the top and bottom terminal strips; they are marked with letters “A” and “B” and also with the up and down arrows that remind of their intended function. Further details can be found in following chapters.

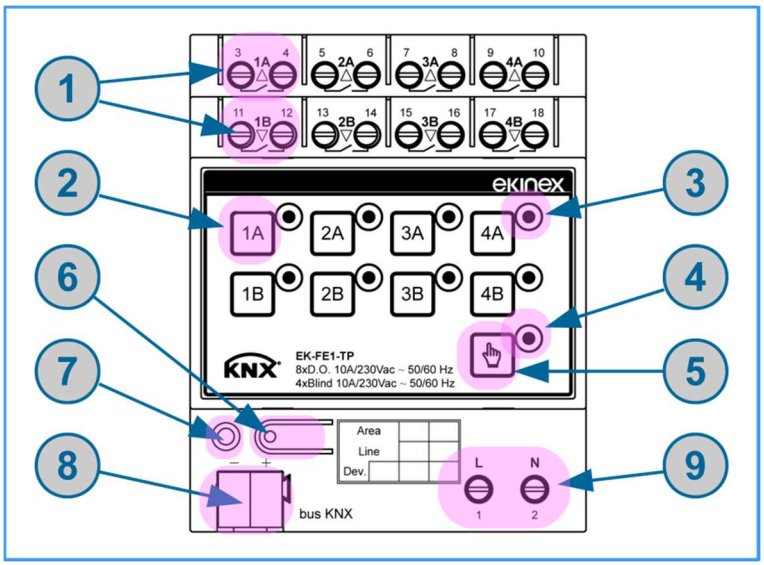


Fig. 1 - Switching, display and connection elements

1) Terminal blocks for outputs.	6) Programming pushbutton
2) Membrane keys for manual operation	7) Programming LED
3) Output status indicator LED	8) Terminal block for KNX bus line
4) Manual mode indicator LED	9) Terminals for auxiliary power supply
5) Membrane keys for mode switching	

4 Configuration

The exact functionality of the device depends on the software settings.

In order to configure and commission the device you need ETS4 or later releases and the proper ekinex® application program, either APEKFE1TP.knxprod or APEKFF1TP.knxprod, which can be downloaded from the ekinex® website www.ekinex.com.

The application program allows the configuration of all working parameters for the device.

The device-specific application program has to be loaded into ETS or, as alternative, the whole ekinex® product database can be loaded; at this point, all the instances of the selected device type can be added to the project.

For every single device, ETS allows to set the operating parameters individually for each input as described in detail in the following chapters.

The configuration can, and usually will, be performed completely offline; the actual transfer of the programmed configuration to the device takes place in the commissioning phase.

Product code	EAN	No. of channels	ETS application software (## = release)	Communication objects (max nr.)	Group adresses (max nr.)
EK-FE1-TP	8018417181177	4/8	APEKFE1TP##.knxprod	222	254
EK-FF1-TP	8018417181184	8/16	APEKFF1TP##.knxprod	442	254

Configuration and commissioning of KNX devices require specialized skills. To acquire these skills, you should attend training courses at a training center certified by KNX.

For further information: www.knx.org

5 Commissioning

After the device has been configured within the ETS project according to user requirements, the commissioning of the device requires the following activities:

- electrically connect the device, as described in the product datasheet, to the bus line on the final network or through a purposely setup network for programming;
- apply power to the bus;
- switch the device operation to programming mode by pressing the programming pushbutton located on the front side of the housing. In this mode of operation, the programming LED is turned on steady;
- upload the configuration (including the physical address) to the device with the ETS program.

At the end of the upload, the operation of the device automatically returns to normal mode; in this mode the programming LED is turned off. Now the device is programmed and ready for use on the bus.

6 Function description

The device is a switching endpoint, which activates its switch channels according to telegrams sent by other devices on the bus.

It also incorporates additional features such as e.g. timing and logic combination features, described in the following chapters.

The outputs are of binary type, i.e. they can only be switched On or Off; each output has a relay with a single-pole, single-throw contact rated 10 A at 230 V AC.

6.1 Startup

After switching on the bus, the device becomes fully functional after a very short time needed for reinitialization. A further delay is programmable for the device to become active on the bus in order to avoid a bus traffic overload during the first moments of startup of the whole network. Assumed that the auxiliary power supply is already applied (or otherwise as soon as it is applied), the device is then ready for operation.

6.2 Offline operation

The device has limited operation capabilities also if one of its two power sources should be missing, i.e. the auxiliary 230 V AC power source or the KNX bus supply.

The internal circuit part that handles communication and logic management can take its supply from any of the two available sources; the power for relay switching, for consumption reasons, is only taken from the auxiliary power supply.

Of course, when both power sources are missing, the device is effectively off.

6.2.1 Operation with bus supply only

In absence of the auxiliary power supply, all functions of the device are effective up to (and including) the determination of the status of the outputs, including feedback; the actual switching of the output relay contacts does not take place though.

In order to detect this probably undesired situation, a power-off alarm communication object can be enabled, so other devices on the bus are able to take proper measures and/or signal the anomaly to the user.

To give a visual cue of the lack of auxiliary power, the LEDs on the front panel are set to flash.

6.2.2 Operation with auxiliary supply only

When the bus power supply is not applied, or in case of a bus power failure (voltage lower than 19 V for 1 s or more), the device features suspend: the timing functions are not active anymore. The device can still be switched to manual mode though, and operated in that way.

As soon as the bus voltage is restored, the device will resume operation in its previous state, unless different initialization settings are programmed.

6.2.3 Output restore

For any mode of operation, the status of the device after some significant events can be defined by configuration. These events are:

- Device power on, i.e. after the auxiliary power supply is applied;
- Bus off, i.e. after a KNX bus failure
- Bus on, i.e. after recovery from a KNX bus failure
- Download of a new or updated configuration from ETS

Further events are associated with specific functions such as the Lock or the Forcing functions.

For each of these events, the status of the output (or output pair) can be configured from a set of values that depend on how the output is configured; these sets of values will be listed later in the sections that describe the corresponding functions.

Please notice that, in all above cases, the auxiliary power supply is supposed to be applied, otherwise the output switching could not take place.

6.3 Manual operation

The manual operation works as an alternative to the output switching through bus commands (*bus-controlled* mode) ; this mode is intended for testing or maintenance purposes.

6.3.1 Status of the outputs across modes

Upon entering manual mode, all outputs maintain their current status. When the manual mode is active, any signal changes coming from the bus will not affect the current status of the outputs, and the device can only be operated via the membrane pushbuttons on the front side of the device.

Manual operation does not cause any telegram to be issued on the bus for status change. The LEDs associated to each pushbutton continue to show the status of the physical output.

Upon returning back from manual mode the current status of the outputs is also maintained.

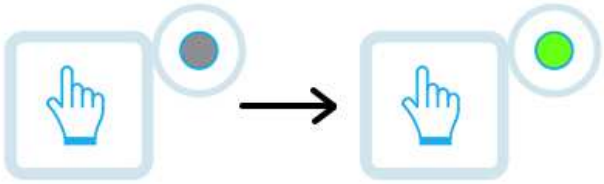
The behavior could be described as if, during manual mode, the internal variables were temporarily “disconnected” from the group addresses; when returning, the variables are “reconnected”, but their content does not change until a new bus command is issued which involves a change in their value.

The same as for bus commands applies to internal timing functions (such as delays and staircase lighting): state changes originated by internal functions do not have effect as long as manual mode is active.

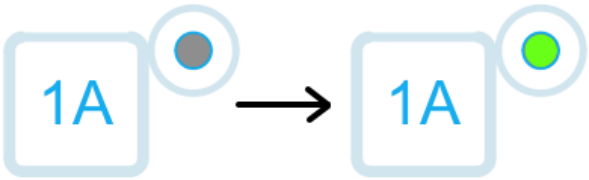
6.3.2 Activation of manual mode

To switch the device to manual operations mode, proceed as follows:

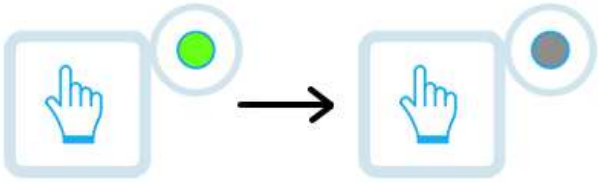
1) Press the manual mode pushbutton. In normal operation the LED is turned off. When the LED turns on, the whole membrane keypad is activated and the manual operations are allowed.



2) Press the pushbutton of the keypad corresponding to the channel that has to be operated (in the example: 1A). Pressing it repeatedly changes the status from On to Off and back.



3) When the required operation is finished, the manual mode is turned off by pressing the mode pushbutton again. Upon returning to bus-controlled mode, the output values will be restored as already described.



Switching to manual mode through the front panel can be inhibited in two ways, both selectable through configuration parameters:

- by disabling the manual switching feature altogether;
- through a bus command.

Please notice for clarity that the bus command mentioned above inhibits switching to manual through the panel key; it does not itself switch modes.

If manual mode is neither inhibited by configuration nor controllable through the bus, another parameter allows to set a timeout period after which, whenever the device is left in manual mode, it will be reverted to bus-controlled mode. This prevents the device to be inadvertently left in an unintended state.

6.4 Online operation

All features described below assume the device has been correspondingly programmed by means of the ETS tool. A fully unprogrammed device causes no activity on the bus; it can be switched to manual mode and operated through the membrane keys on the front panel.

6.4.1 Software working cycle

The software working cycle can be described as follows:

- Handle incoming telegrams from the KNX bus to update internal state variables
- Implement timing functions and other inbuilt functions to determine effect on physical outputs;
- Drive output relays outputs according to output status
- Handle the key presses from the membrane key on the front.
- Respond to bus messages requesting feedback on the status of the outputs and of the device.

There are also special events on which it is possible to trigger additional features. These events are for instance the bus and power supply failure and recovery, and the download of a new configuration with ETS.

6.4.2 State variables (Communication objects)

The determination of the status of physical outputs is made basing on internal state variables. These state variables, once assigned a group address, are actually KNX communication objects, which allows other devices on the bus to exploit the features of the device.

State variables undergo the usual rules for communication objects, among which – for instance – the effect of flags to determine how the change of value affects the transmission of the objects.

6.4.3 Output independent mode and coupling

Outputs can be driven independently, or they can be coupled; the features available in both modes will be explained in detail in following chapters.

Due to the nature of the functions this device most frequently performs, the outputs can be grouped in pairs. In this case, each channel is made of a pair of outputs which are physically close on the terminal block.

In order to maintain a consistent naming, the outputs are numbered in the same way regardless whether the channel pairing is used or not.

The coupled channels of the device are labelled 1 to 4, whereas the outputs are labelled 1A / 1B for channel 1, 2A / 2B for channel 2 and so on; for convenience, this same enumeration is used for labelling even if the outputs are used individually.

In order to specify channel pairings, each output can be configured in two ways: single mode and coupled mode.

- In *single mode*, each output operates independently, has its own parameters and communication objects. This is the mode of operation described so far.
- In *coupled mode*, 2 outputs operate logically grouped under the same channel in order to perform a common functionality. Only outputs belonging to the same channel can be coupled, therefore the only combinations allowed for coupling are 1A with 1B, 2A with 2B, and so on.

It is possible to configure some of the outputs in single mode and the others in coupled mode, with the pairing constraints just described.

6.4.4 Output features in independent mode

In the most simple case there is only one communication object per channel, “On-Off command”, that switches each channel output directly with a message.

By setting the device parameters, it is possible to activate additional features, most of which will also affect the outputs. These features are:

- Relay inversion: allows to short contacts on the Off logical value and disconnect on the On value.
- Feedback: sends message on each switching operation or cyclically each period of time
- Time delay block: allows to perform the actual relay switch with a programmable delay. It is available (with separate delay settings) both for the On-Off and for the Off-On transition.
- Staircase function: performs a retriggerable time period activation of an output.
- Logic function: allows to compute the output value as a logic function based on the value of several communication objects.
- Lock and Force: these functions can temporarily force the output to fixed values and also perform high priority switching operations.
- Scene management: - allows to save and recall a combination of state and values with a single telegram.
- Operating hours / Energy consumption counter: allows a limited tracking of energy consumption by accumulating “On” period durations over time.

The most significant functional blocks for an output in independent operation are described in the following scheme.

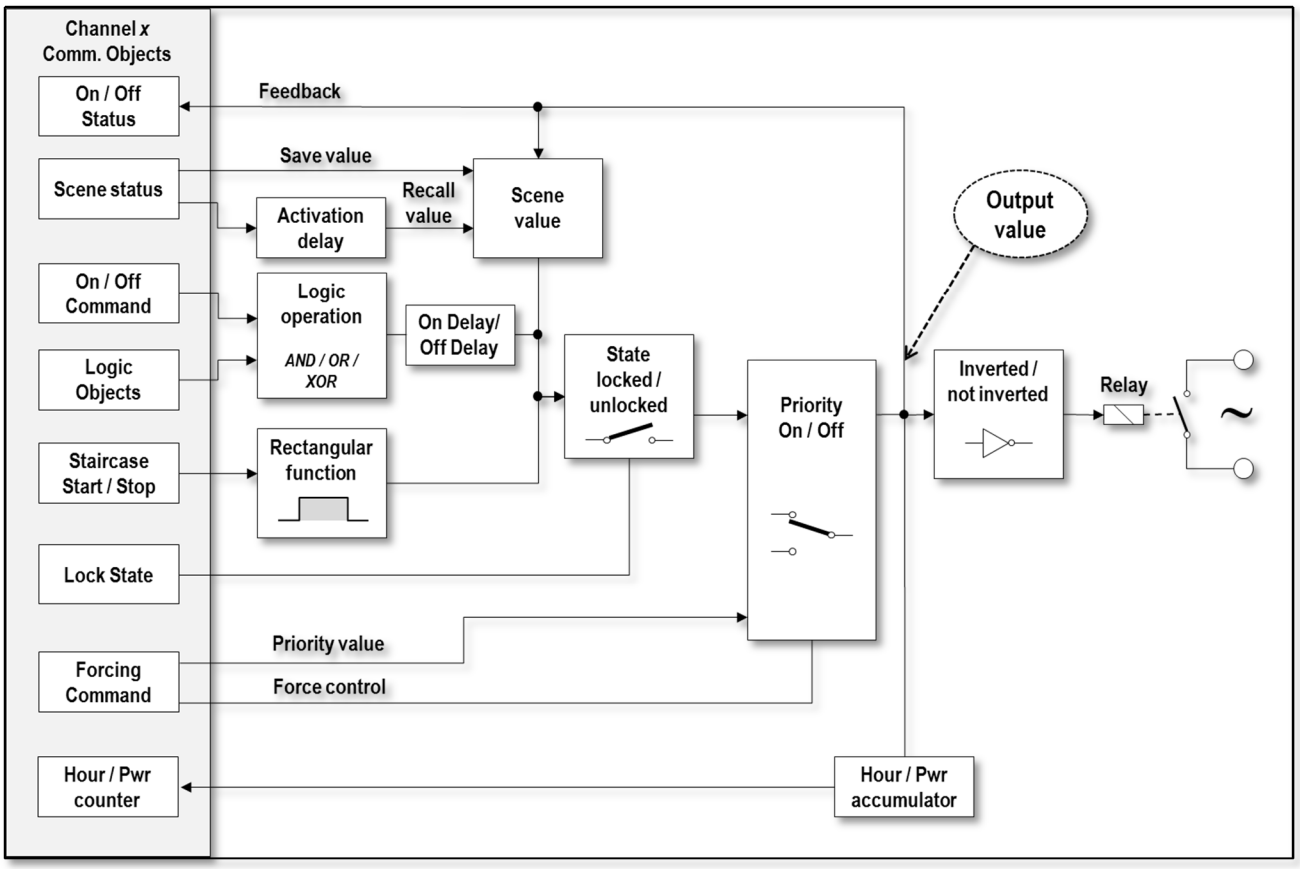


Fig. 2 - Functional blocks – Independent mode (referred to a single output)

It must be noted that, as can be seen from the above diagram, the different features of the output channel can be activated and operated in parallel at the same time; the configurator has the responsibility of taking care that any interference between different functions does not produce unintended effects on the way device outputs are managed.

6.4.4.1 Relay inversion

This feature inverts the status of the physical contact of a channel with respect to the exit status.
Regardless of the “inversion” parameter setting, the following sections will always take “on” and “off” to be a reference to the logical status of the output, not the status of the relay contact switch.

6.4.4.2 Feedback

When feedback is enabled, a communication object corresponding to the status of the output is made available for reading by other devices on the bus. This object carries the actual state of the logic output, which is likely to be different from the command value because it includes the effect of all additional functions which may be active at the time.

If this communication object is defined, it is also transmitted on every state change, so it can be used to trigger events following the actual state change of an output; it is also possible to configure transmission at regular intervals.

Feedback telegrams are not sent if the outputs are operated manually.

6.4.4.3 Time delay

The actual change of state of an output can be set to take place after a configurable delay from the change of the value of the corresponding communication object; this applies both to the on-off and the off-on transitions, each with its individually configurable delay value (T_{on} and T_{off} respectively).

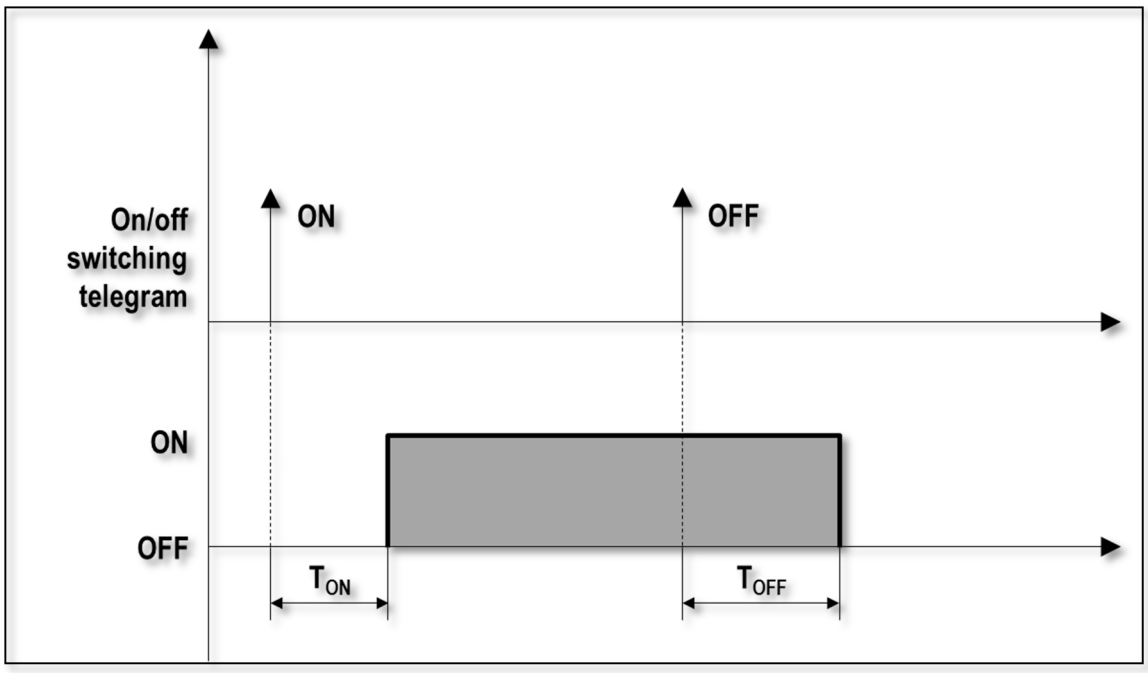


Fig. 3 - Time delay

6.4.4.4 Staircase function

This function is intended to provide a simple and flexible way to manage the switching of staircase lights. These have following peculiar requirements:

- The light is activated by a “start” command (e.g. through a pushbutton or a presence sensor), and normally remain lit for a programmed time duration;
- There is a provision to enable a “stop” (Manual Off) command, again through a pushbutton or other events, that allows to switch the light off before the programmed time expires (e.g. because the person who triggered the presence sensor has surely left the building through an exit);
- There is a provision to allow another “start” command (Retriggering), received during activation, to restart the time duration counter;
- A further optional “pre-warning” function allows to briefly switch off the load a certain time before expiration (both times, i.e. pause duration and time before expiration, are configurable) in order to warn the user that the activation time is about to end.

Following pictures show the *Manual Off* feature:

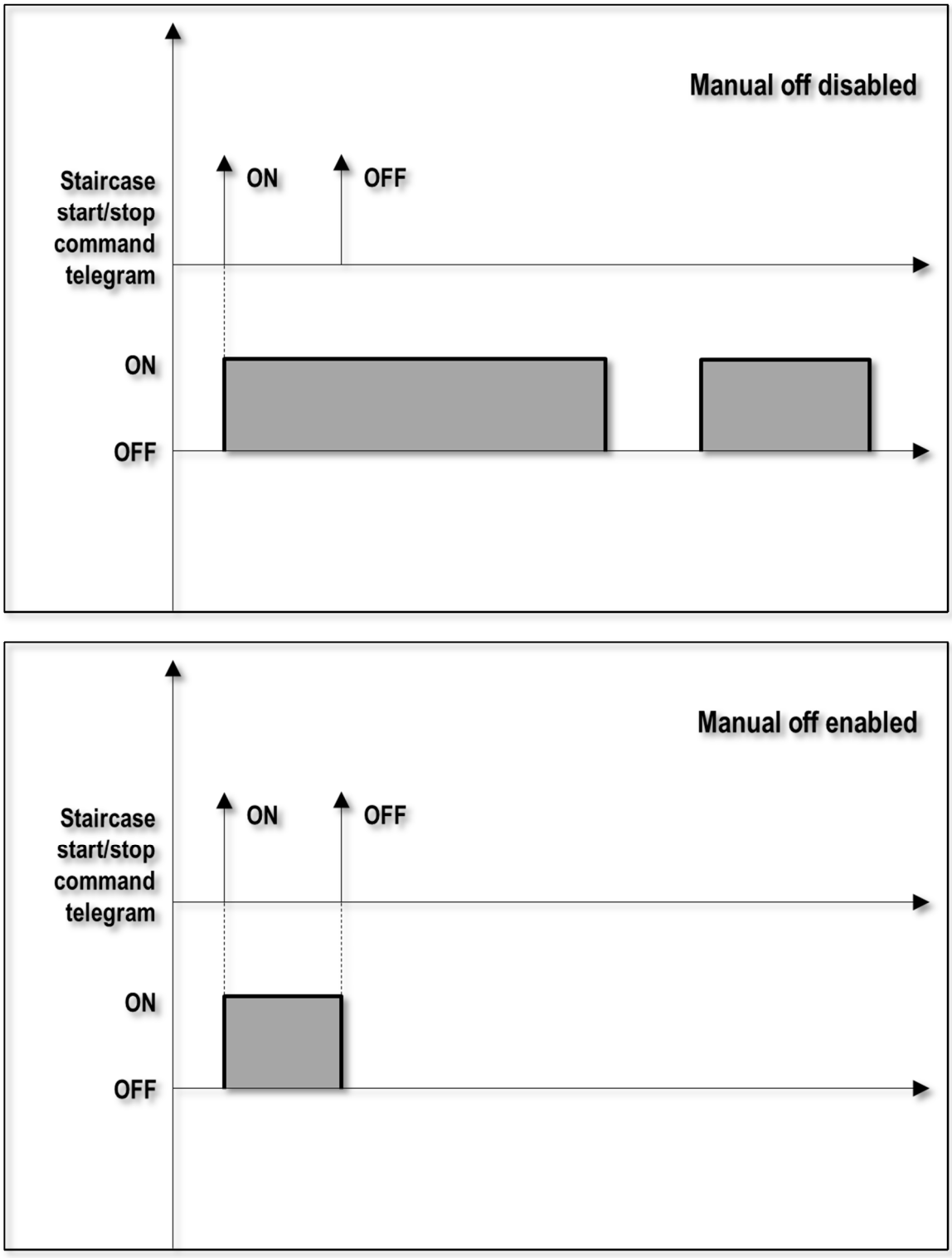


Fig. 4 - Manual Off feature

Following pictures show the *Retrigger* feature:

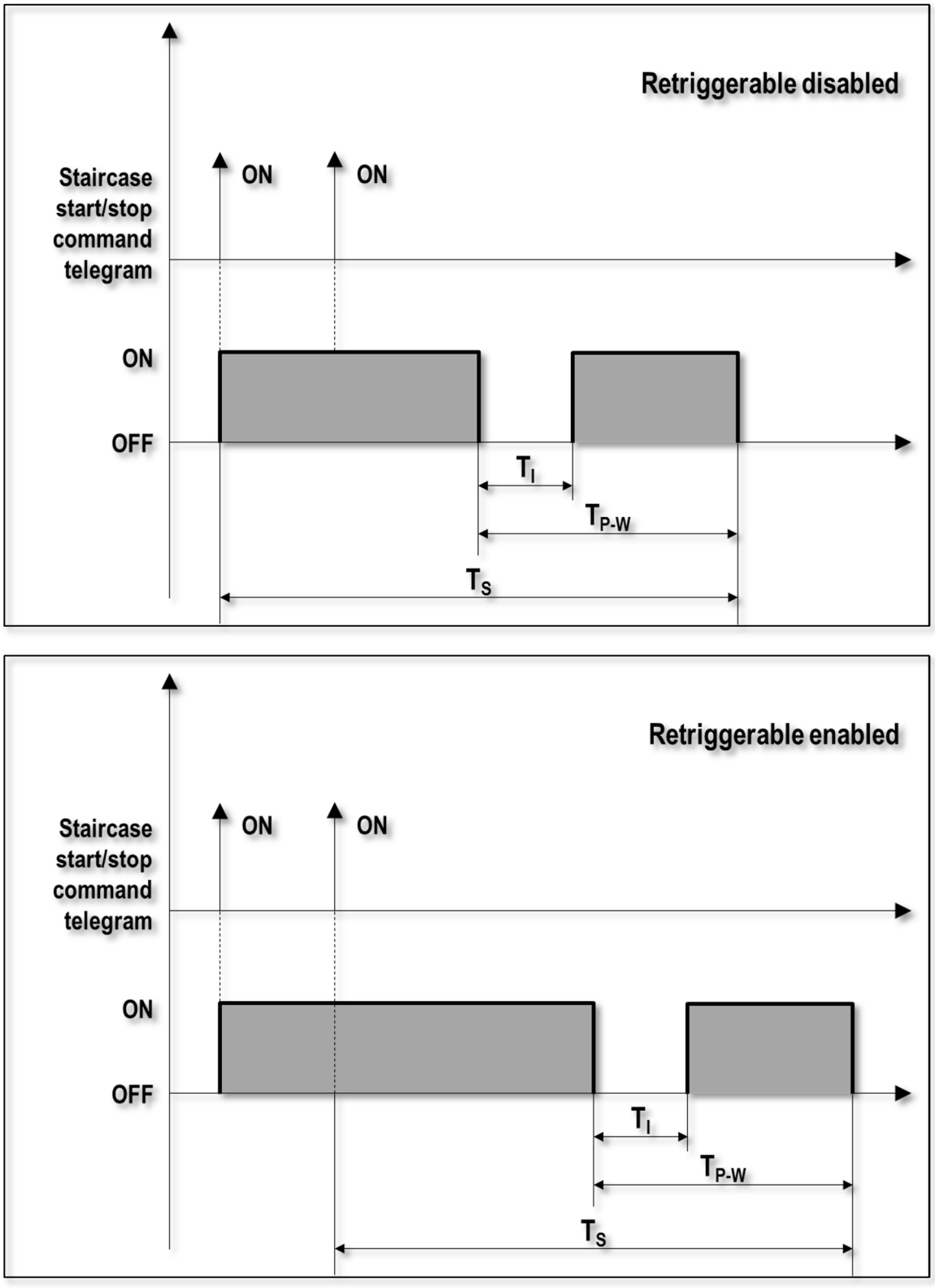


Fig. 5 - Retrigger feature

Following pictures show the *Pre-warning* feature:

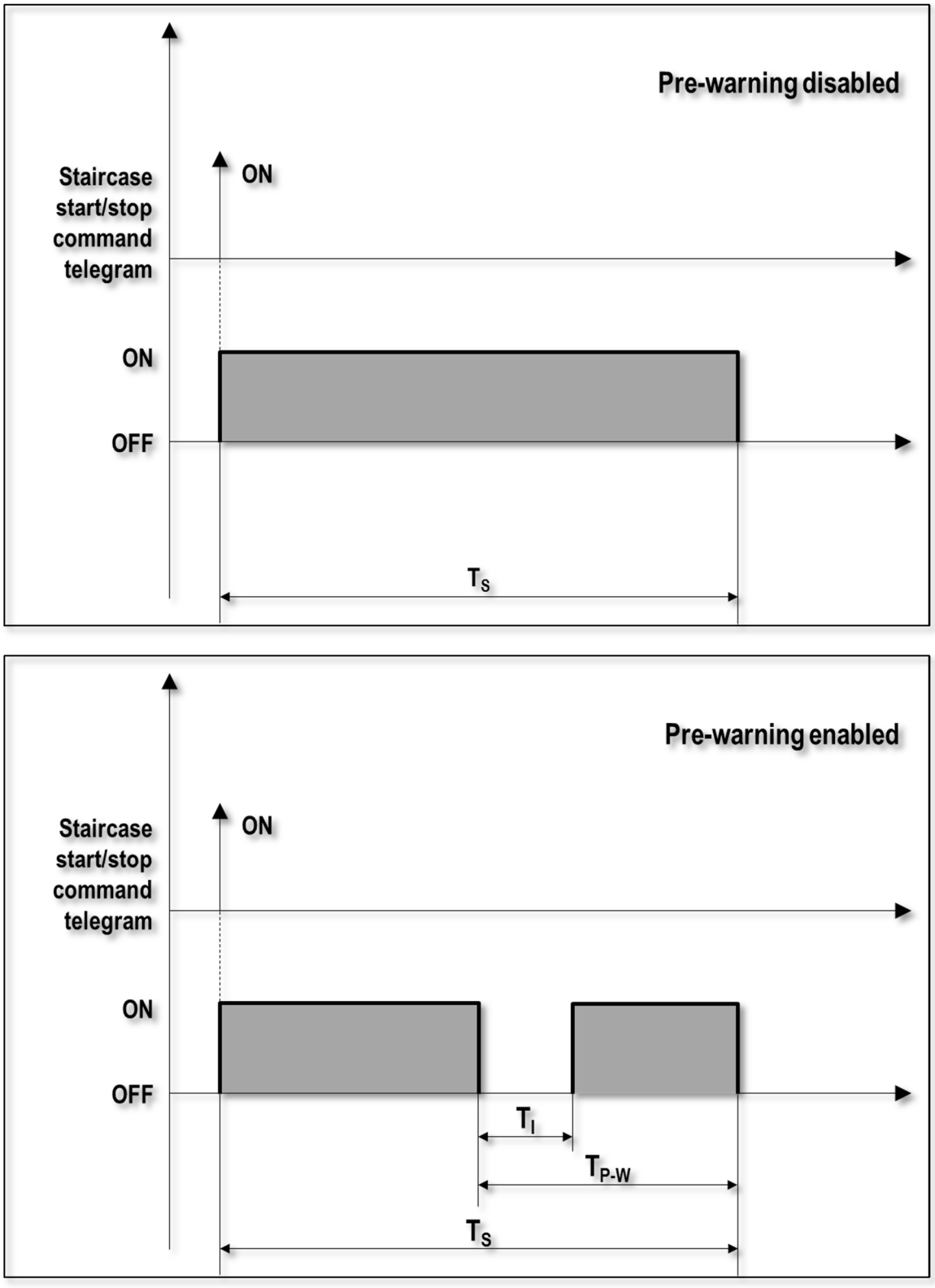


Fig. 6 - Pre-warning feature

6.4.4.5 Logic function

The device has a limited provision for the logic processing of internal variables in order to condition the status of outputs.

A given incoming output command can be used as an input to a logic block which operation is selectable between OR, AND and XOR (exclusive OR). Up to other 8 objects can be defined as additional inputs to the same block (each with an optional negation operation); these objects are directly accessible to other devices from the bus and they can be used as desired.

The input objects are logically combined as in following picture:

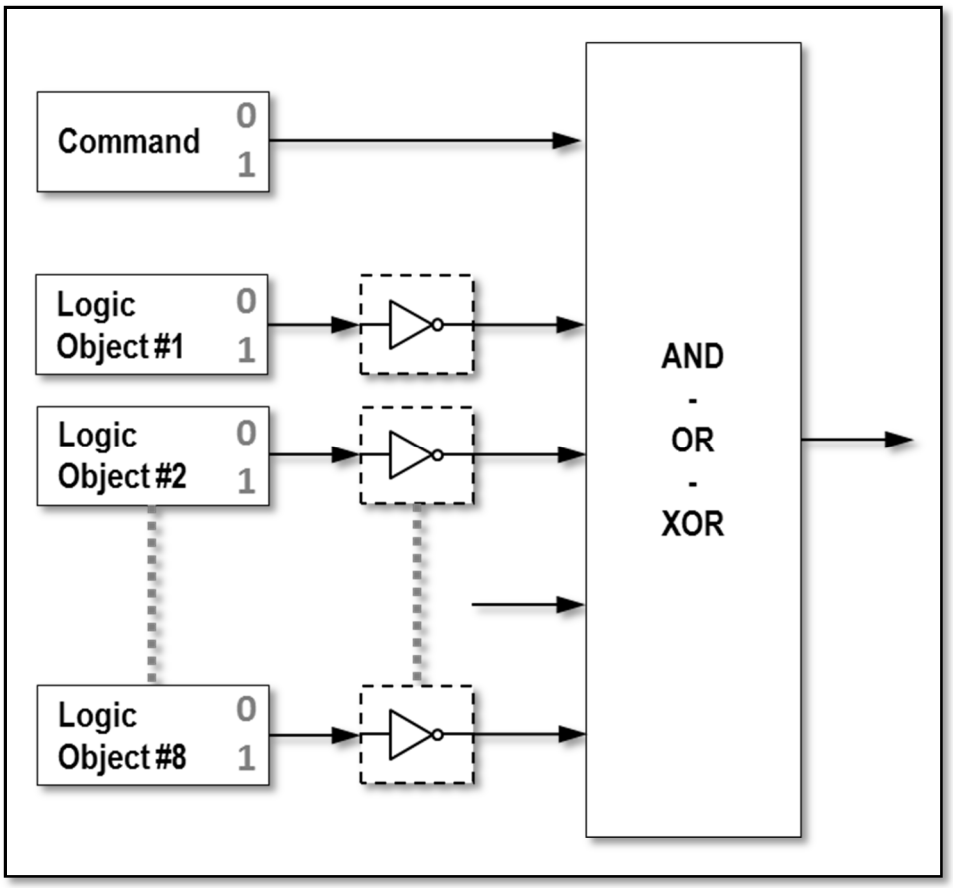


Fig. 7 - Logic functions

The logic combination block on the right works as follow according to which logical operation is selected:

- OR – the output is ON whenever any one of the inputs is ON;
- AND – the output is ON only if all of the inputs are ON;
- XOR – the output is ON if an ODD number of inputs are ON.
  This latter operation is more intuitive when thinking of two inputs only: in this case, the output is ON when one input or the other is ON, but not both.

It must be noted that, in the above description, “input” and “output” are referred to the logical block; for the purpose of operation, the actual “inputs” are the logic objects, thus the optional inverters must be factored in.

This structure allows to implement fairly complex logical combinations; a more generic and powerful programming capability would add more complexity and therefore it would be far beyond the scope of an output module that is simple to use.
In the following pictures, the basic logic functions are illustrated, assuming the output command and one logic object are used:

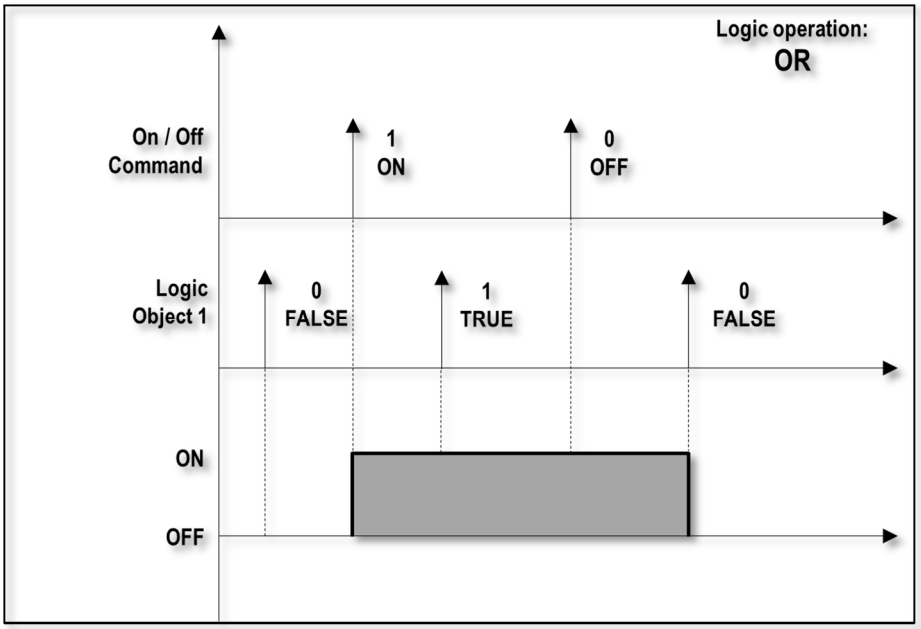


Fig. 8 - Logic OR function

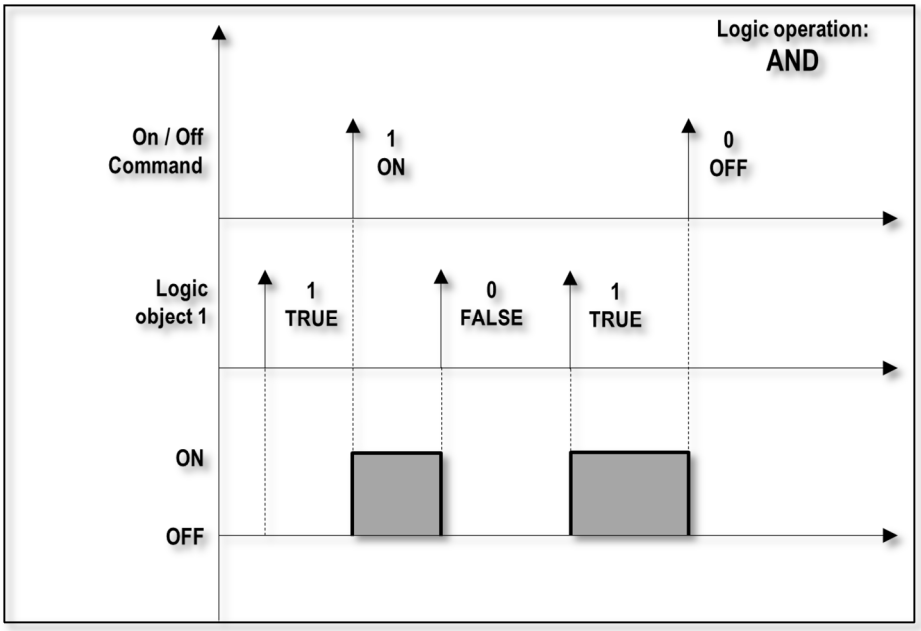


Fig. 9 - Logic AND function

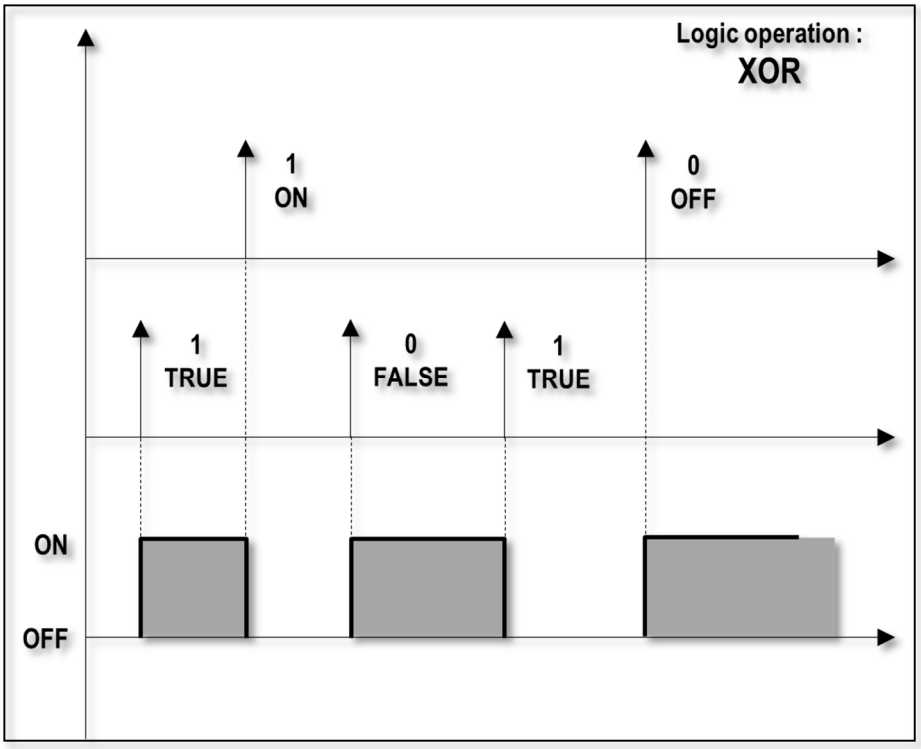


Fig. 10 - Logic XOR function

6.4.4.6 Lock function

If the locking feature is enabled, the operation of a channel can be inhibited by writing a value in a communication object. The value written is of the KNX type “enable”; please beware that the meaning of this value is “*activate lock*”, which is not to be confused either with “enable *locking function*” or with “enable output”. The meaning of the value can be optionally inverted through a configuration parameter (an “enable on” value can be interpreted as “lock off”).

A locked output ignores the switching commands that are received for the duration of the lock, thereby maintaining the status it has upon lock entry. The status of the output can be set to a particular value both when the lock is set and when it is released; it is also possible to determine whether the lock status should be maintained or changed on recovery after a bus power-off.

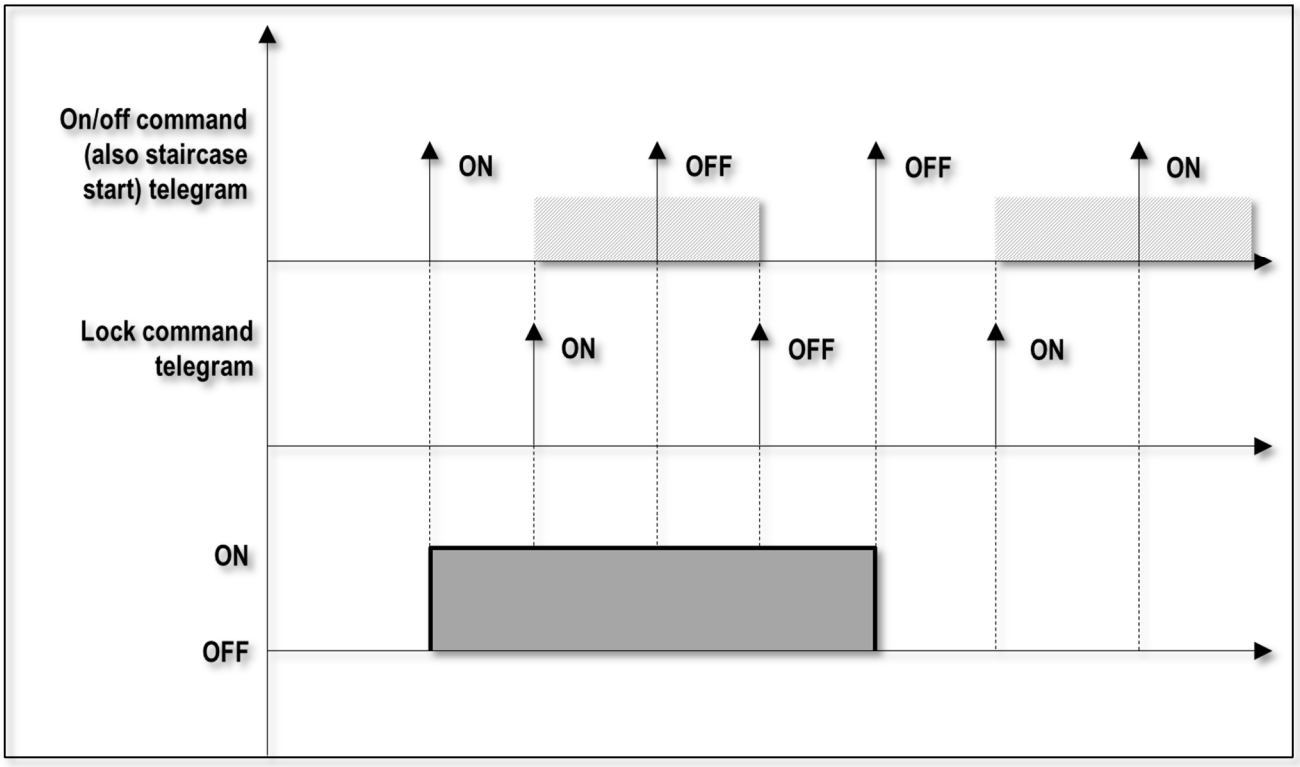


Fig. 11 - Lock function

6.4.4.7 Forcing function

The forced control is very similar to the basic direct command of the output value, but with the peculiarity that it overrides both the “regular” set value and every other value conditioning feature (i.e. logic function, staircase timing etc.).

It is possible to set what value the output should assume both when the output forcing is released and also on recovery after a bus power-off if forcing was previously in effect.

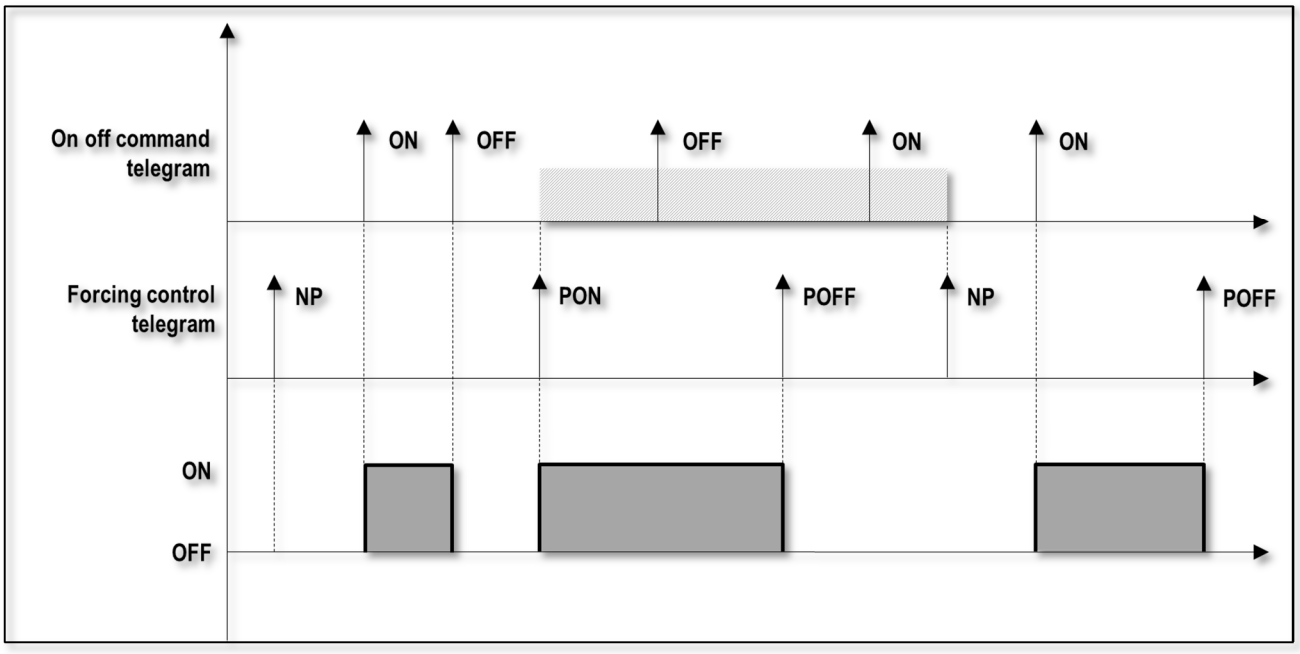


Fig. 12 - Forcing function

The “Force” command has priority over Locking (which acts on the ordinary on-off command); therefore, a locked output can still be operated through “Force” commands.

The KNX command code for the “Force” operation is a 2 bit value; the *priority* bit determines whether the output value must be forced, in which case the *value* bit is assigned to the output.

In the figure above, NP means that the *priority* bit is 0 (No Priority), while the PON and POFF codes indicate the values with *priority* = 1 and *value* respectively 1 or 0.

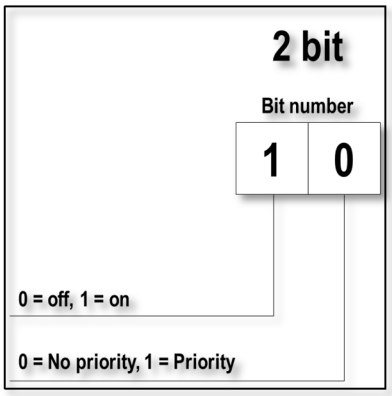


Fig. 13 - Force command bits

6.4.4.8 Scene management

Each output can be linked to up to 8 scene codes; when one of these scene codes is recalled through a bus command originated by any controller device, the output will assume a preset value. An additional delay can be defined for the output activation (or deactivation) from the moment the scene code is recalled.

The output value for a scene can either be fixed and chosen in the configuration phase, or it can be defined as reprogrammable through a Scene Learning command.

If this latter option is enabled (for each single output), whenever a Scene Learning command is received on the bus for a specific scene code to which the output has an association, the device will store the current output status value for that scene. This value will then be recalled in subsequent scene activations.

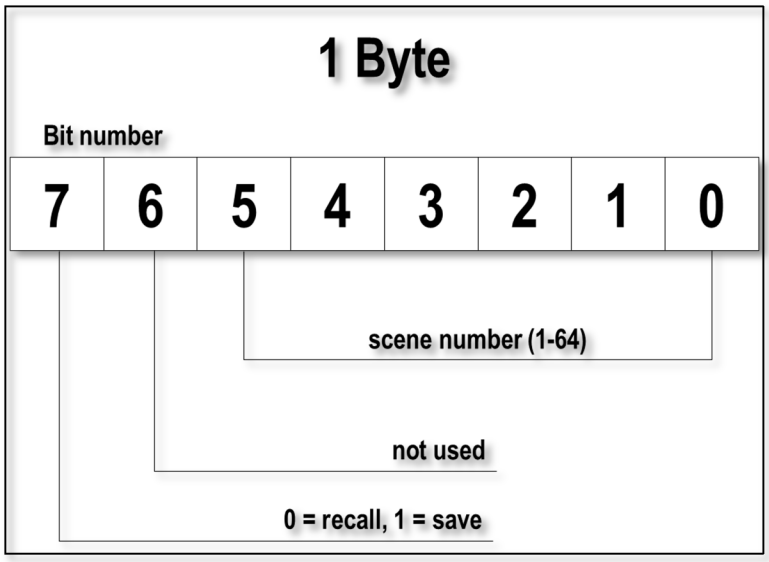


Fig. 14 - Scene store / recall command code

6.4.4.9 Operating hours / Energy consumption counter

For each output, an activation counter can be associated which accumulates the count of hours that the output passed in the “on” state. In terms of communication objects, this counter has the format of a KNX hour counter, thus it also has a “reset” command ad a “runout” alarm in case the maximum value is overflowed.

An additional parameter allows to define a conventional electrical power which is associated to the load; although this is not a “real” power metering, but merely a conversion factor between activation time and the estimated consumed power, nonetheless it can supply a useful indication for approximate power monitoring, particularly for resistive or fixed-power loads like lights or many other home or office appliances .

The power counter also has an associated KNX “kWh counter” communication object with its own reset command.

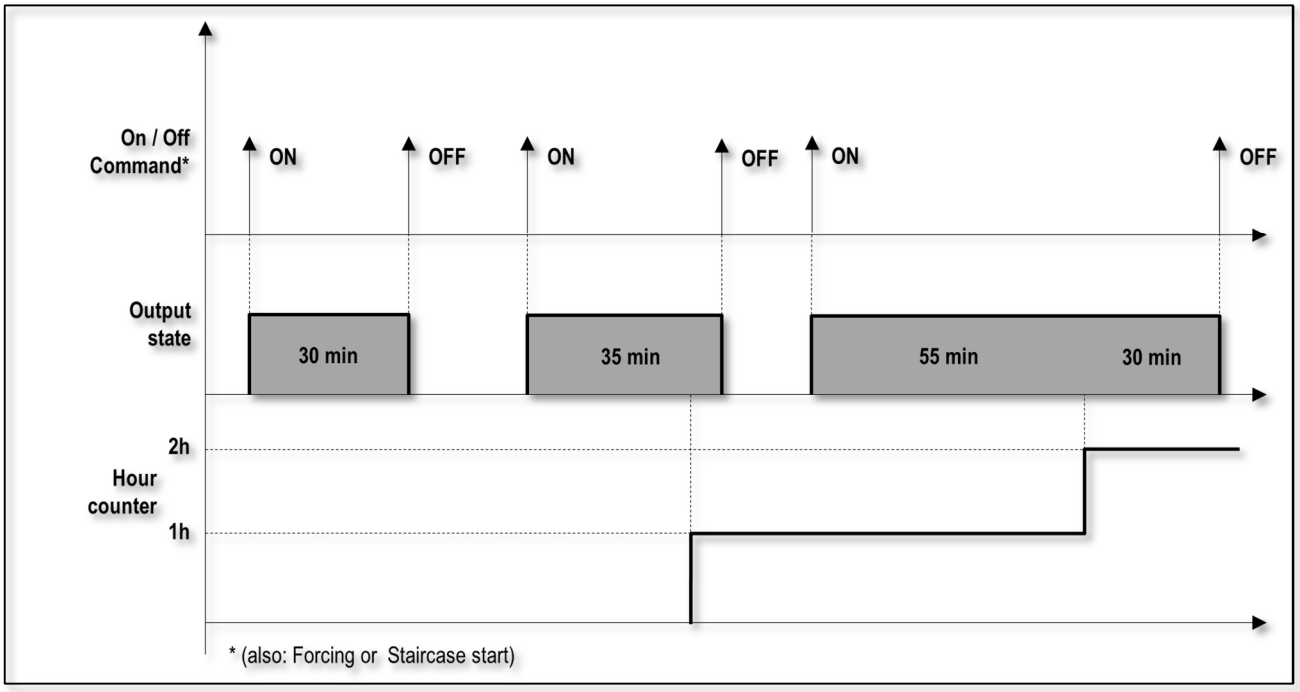


Fig. 15 – Operating hours and energy counter

6.4.4.10 Output restore values

As mentioned in an earlier paragraph, the status of the device after some significant events (see “*Output restore”* paragraph for description) can be defined by configuration.

The values available for restore after system events for independent inputs are:

- On
- Off
- no change
- previous value / state*
  (* this option is not available for either “bus off” or “after download” events).

The difference between “no change” and “previous state” is following:

- “no change” refers to before the event itself (e.g. for the “bus on” event, an output which was “off” before bus recovery will remain “off” thereafter);
- “previous state” refers to before the condition that is terminated by the event (e.g. for the “bus on” event, an output which was “on” before bus failure will return “on” after bus recovery).

6.4.5 Output features in coupled mode

In coupled mode, output pairs can be used to drive three categories of devices: these are grouped under the denomination of *Valve actuators* (2- or 3-way), *Shutters* and *Venetian Blinds.*

These categories have basically a similar operation mode, that is, they move a physical device from one to another endpoint; this can happen stepwise, with full stroke, or possibly stopping at given intermediate positions. The mentioned actuators, in the order they are listed, could be seen – apart from minor details - an increasingly sophisticated version of the same basic mechanism. Anyway, all three of them are driven through two lines, one for each direction.

For any single channel, one of these three types of behavior can be chosen.

Beside the distinctive features of these categories, there are further features common to all of them, like the locking and forcing functions, meteo alarms and scene management, that will be described below. Some of these features are similar to those described for those of single outputs in independent mode; in these cases, the corresponding sections in the previous paragraph are referenced.

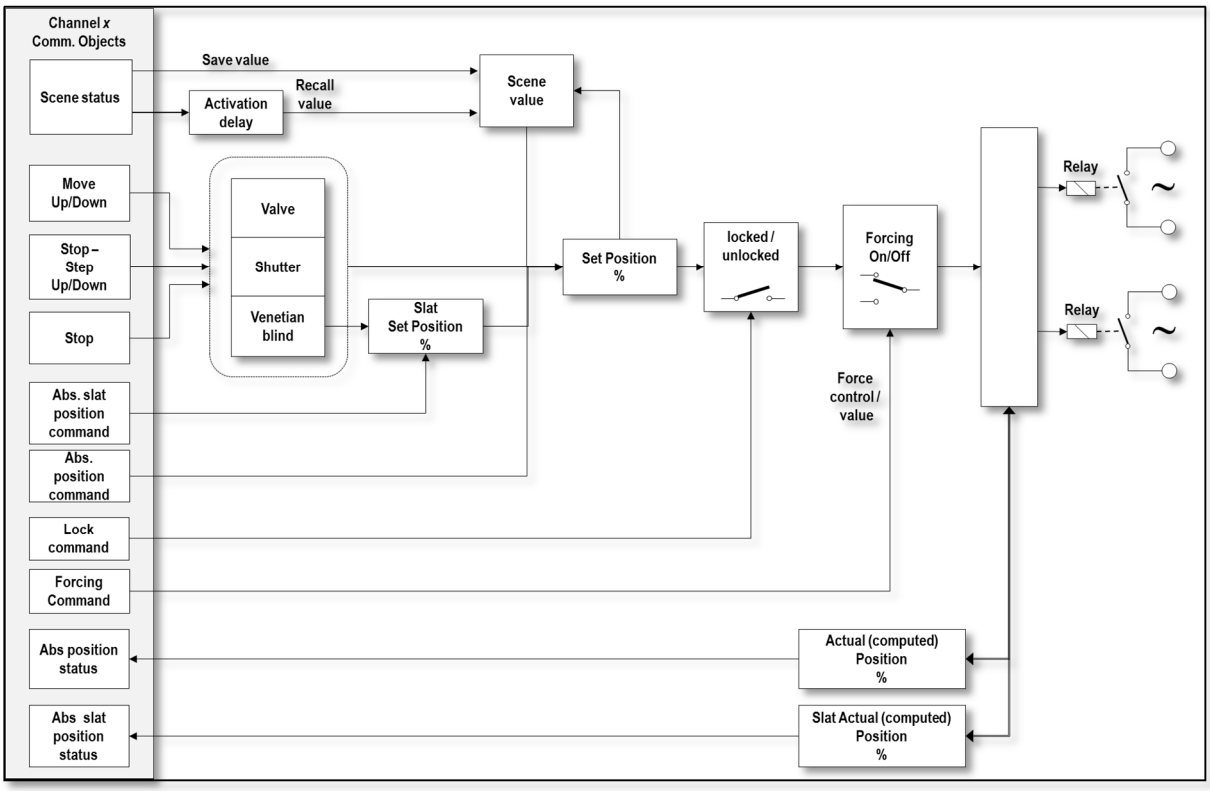


Fig. 16 - Functional blocks – Coupled mode (referred to a single output)

6.4.5.1 Coupled output control basics

The control with coupled outputs is based on three main telegrams, all of which are 1-bit values and thus can convey up to two commands each:

Move Up (Open) / Down (Close)	When the telegram is received, the actuator starts moving all the way towards the specified endpoint.
Dedicated stop	When the telegram is received, the actuator stops any movement and remains in the current position
Stop – Step Up / Down	This command allows a gradual or stepwise movement of the actuator. It actually has a dual purpose: • when the actuator is at rest, it acts similarly to the Move Up/Down command. When the telegram is received, the actuator moves in the specified direction, but just by one “step” (i.e. a length predefined by timing); • when the actuator is moving, it stops in the current position.

In most actual systems, as also defined by KNX standards, the difference between “Move” and “Step” (aside from the additional “Stop” function of the latter) is just the length of the time interval: in principle, a “Move” command is just a “Step” command which duration is guaranteed to be long enough to allow the actuator to reach the endpoint.

Looking at it another way, the same timing that in the case of stepping defines the Step duration, in the case of the Move command has the role of a timeout that deactivates the output when it is no longer necessary to drive it. (Of course there are different parameters for these timings). Actuators, anyway, will normally have electrical end switches that will prevent overloads caused by unnecessarily applying power when at the endpoints.

Since no position feedback is available from the mechanical actuator, the shutter position is determined through movement timing: given the full-scale movement time value (i.e. the exact time the shutter / actuator takes to move from one endpoint to the other), a partial movement expressed in a percent fraction of the full stroke will then correspond to the same fraction of movement time. The device keeps an internal position counter which is realigned whenever a full Move up/down command is issued.

In order to have the correct timing to be applied to output switches, the full-scale movement time value must be set through a parameter.

This is just a basic generic description; actual actuator types may not have the same control possibilities (e.g. they might not be capable of stopping in positions other than the two endpoints) or they may have more options and features. This will be described below in the explanation of specific functions.

6.4.5.2 Valve control

The valve control is the most basic of the three controls available; the control can be configured for both 2- and 3-way actuators.

A 2-way actuator has two command lines: one line brings the valve in one (say “open”) position, while the other moves it the opposite way. There are no intermediate rest positions.

A 3-way actuator works almost the same way, except that the movement between the two endpoints is gradual (and slower); therefore, if both command lines are de-energized while the actuator is travelling between the endpoints, it will stop in the current intermediate rest position.

Since a 3-way actuator works exactly like a Shutter control, which is described in the next section, only the 2-way actuator will be described here.

This control supplies the three basic commands already described in the “basics” section; however, the “Stop/Step” command is provided because it is required by KNX specifications, but it has no practical effect because no gradual movement is possible. The Stop command also has no practical effect on the movement (other than de-energizing both outputs immediately).

The standard way of driving a 2-way valve requires therefore just the “Move” command to be issued with either direction set in order to switch the valve to either position.

An additional communication object is available to query the movement status of the actuator (i.e. it indicates whether the valve is moving or at rest).

6.4.5.3 Shutter control

The shutter control is the most similar to the typical control described in the “basics” section; the description of its operation also applies exactly to the 3-way valve.

This control supplies the three basic commands already described in the “basics” section; however, the “Stop/Step” command is provided because it is required by KNX specifications, but only acts when used as a “Stop” command (it has no effect when the actuator is not moving).

The standard way of driving a shutter channel is therefore the following:

- issue the “Move” command with either direction set, in order to start the motion of the shutter;
- either leave the shutter to arrive to the endpoint (the output will be deactivated after a timeout anyway, see below) or issue either a “Stop” or a “Step/Stop” command as soon as the shutter has reached the desired intermediate position.

In order to better exploit the possibility of intermediate positioning, this control has additional ways to specify the actuator position:

- the position can be specified as “absolute position” (in percentage); a feedback value for the actual current position and a telegram of “valid position” (setpoint reached) are also available;
- if enabled, a dimmer-type control for the position is also available, as illustrated in figure below. Please refer to the parameter description section for more details.

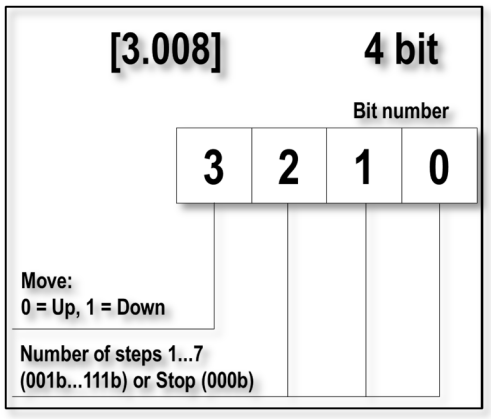


Fig. 17 - Dimmer-type blind control

As already mentioned, the full-scale movement time value must be set; there are two parameters for this purpose, one for the upward and one for the downward direction. Times in two directions may be different for mechanical reasons (e.g. heavy shutters) or functional reasons.

The time amount to be specified is the actual and exact stroke time frome one endpoint to another; this will be used to compute the timings for the requested movement stretches. If a movement must be effected that guarantees that the endpoint is surely reached, its duration will be set to 120% of the specified value.

Another parameter which must be defined for the shutter movement is the reversion pause time, i.e. a pause to be made when a movement command in one direction is issued while the shutter is moving the opposite direction. This is mainly made to allow the shutter to correctly stop without excessive strain on mechanical organs.

6.4.5.4 Venetian blind control

The Venetian Blind has the same features as the Shutter control, but with a few additional parameters dedicated to the management of slats (or louvers).

In terms of available commands and parameters, Venetian blinds differ from Shutters in following respects:

- the “Step” command is now meaningful. A step movement is referred to the slats (not to the blinds panel opening); there is a corresponding parameter to define the step time, i.e. the activation time for the outputs corresponding to the movement of a desired step;
- a further set of communication object for “absolute position”, “absolute position status” and “Valid position” is available for slats;
- a further dimmer-type control is also available for the slats.

Since slats also have their own absolute positioning feature, a parameter for the total movement time of the slats, similar to the one defined for the blinds, is also provided (but in this case common to both directions, since little or no mechanical asymmetry is to be expected). An internal position counter, similar to the one for the blinds or shutter position, is managed to guarantee the best possible precision in positioning.

Standard blinds’ actuators control both blind and slat movement through only two interface lines, the same as shutters discussed in previous paragraph; in order to achieve control of both movements, they are driven as described below. Please bear in mind that this is a principle description of a simplified, albeit realistic, mechanism just for illustration purpose; actual devices may employ different or more sophisticated solutions to realize the same functionalities.

As a general description, each of the driving lines (for respectively upward and downward movement) of the actuator motor directly moves the blind panel towards the corresponding direction. In doing so, the slats are

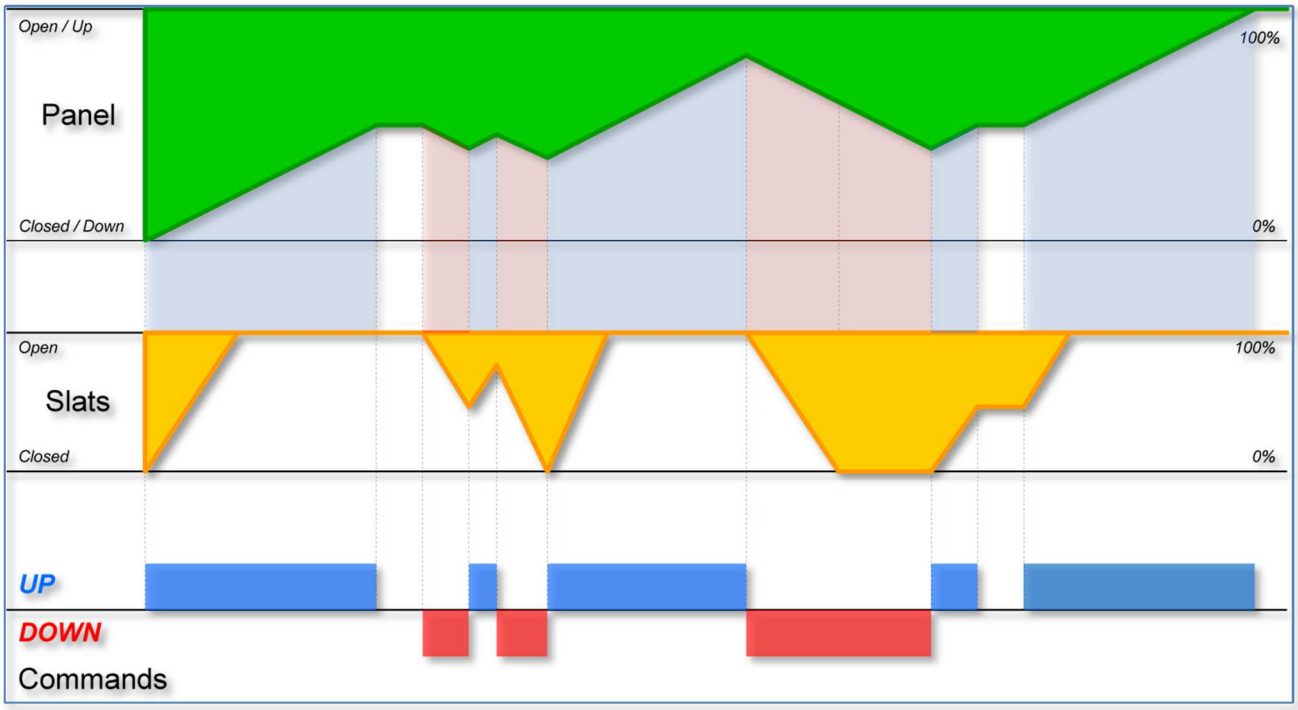
"dragged" in the same direction as the panel (i.e. opening or closing) until they reach their fully open or fully closed position.

We first assume that the blinds start in fully closed position. Activating the "open" line, the motor starts to drag the blinds' array upwards; the slats also move towards the open position. Once these have reached their endpoint, the further action of the motor just continues to lift the blinds.

Assuming now that the blind is stopped halfway, we have a partially open blind with fully open slats; we may naturally continue from here all the way until fully open. If we now activate the downward driver line, though, the slats are moved towards the closed position while the blinds' panel begins to move. The slats are eventually fully closed and the blinds continue to move downwards.

If the activation time of the downward driver line was brief, i.e. not long enough to have the slats span all the way to the closed position, we would obtain a situation where the blind has moved down slightly, but the slats are in an intermediate position; in fact, by alternating the activation of the up / down lines, they can be brought in any desired intermediate position

The following picture illustrates how the blinds react to a command sequence:



As apparent from the description above, the slats cannot be moved independently from the blinds' array, i.e. small drive pulses do move the slats as desired but also modify the blinds' position slightly. In order to compensate for this effect and achieve a slat movement without changing the blinds' position (unless temporarily), a "recovery" movement is effected, much like the backlash recovery in automated tools.

This recovery works as follows. Let's assume for example that we would like to lower (close) the slats starting from a 50% position to a 70% position. When the downward line is activated, the blinds' panel is also lowered a little (length "L1" in the picture below). The actual movement is therefore corrected as illustrated in the second part of the picture (which is shown from the original starting position for clarity's sake).

The blinds are initially raised until the slats are fully open (length L2), and then further to compensate for the mentioned length L1. After that, the downward line is activated for as long as necessary to bring the slats to their desired position. The final result is as intended.

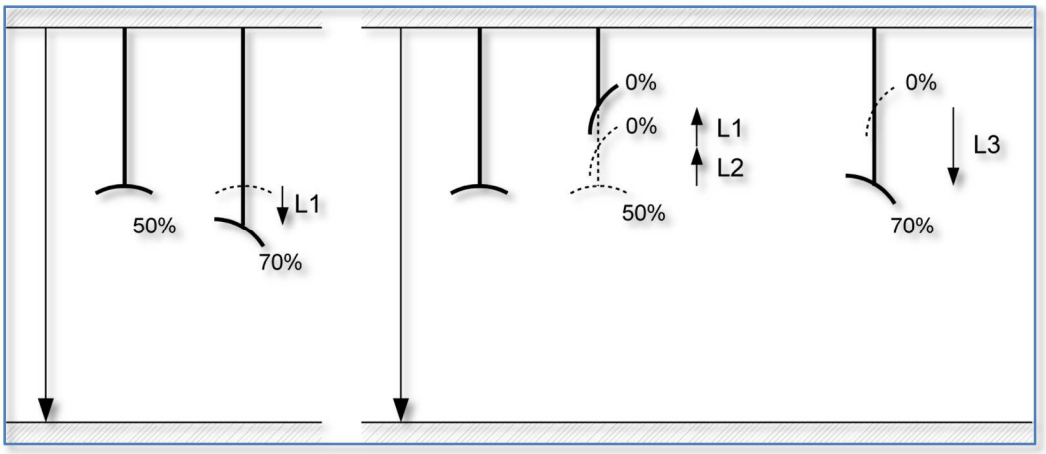


Fig. 18 - Compensation for slat movement

All the lengths (and corresponding movement times) are computed by the device according to the defined time values for full-range movement times for both slats and blinds' panel; both of these times must be configured for the actuator in use as precisely as possible. The compensation mechanism is automatically managed and does not need being accounted for either by the configurator or the final user.

6.4.5.5 Lock function

The locking feature is similar to the case of independent inputs; the only actual difference is in the wider range of values that can be assigned to the actuator position with respect to simple binary outputs. In particular, these values include stopping current motion, moving the actuator to one of the endpoints, to a programmed position or to the position the actuator had before locking.

Further details can be found in the configuration section.

6.4.5.6 Forcing function

The forced control is basically similar to the case of independent inputs; the very same considerations apply as for the case of the Lock function.

6.4.5.7 Meteo alarms

The Meteo Alarms allow to pre-program an actuator deployment in case of meteorological events detected by a meteo sensor unit (which must be separately purchased and interfaced).

Three types of Meteo alarms can be handled independently, namely for Wind, Frost and Rain. The name is actually just descriptive, since the three alarms are perfectly equivalent and can be used even for different events altogether.

For each of these alarms, a behavior can be defined for the actuator when the alarm is received (go to full "up / open" position, go to full "down / closed" position, or do nothing). Another behavior can be associated to the ceasing of all alarms (all choices above, plus return in the state the actuator had before the alarm).

If more than one alarm becomes active, the action associated to the latest alarm is performed in turn.

A KNX alarm has an optional "heartbeat" function, i.e. the telegram associated with the alarm can possibly be repeated (and usually it is) at regular intervals; this has a double purpose, in that it assures that an active alarm is not missed if a telegram is lost for whatever reason, and it also confirms that the alarm source is "alive" and that no alarm condition is active if this is the case (alarm telegrams are transmitted with an "Alarm condition clear" value even if the alarm is not active).

For each of the three available alarms, a timeout can be defined for the heartbeat function; if an Alarm information telegram is not received within the timeout duration, the alarm is assumed active and the actuator is correspondingly set. A timeout which occurs when the alarm is already active has no effect.

The heartbeat timeout can of course be disabled; it is important to mention, though, that if it is enabled the device that originates the alarm must be configured for the periodic transmission of alarm information telegrams (furthermore with a period compatible with the timeout interval).

6.4.5.8 Scene management

Scene management function is similar to the case of independent inputs; the same considerations apply as for the case of the Lock function. The values that can be assigned to a scene are the two endpoints, a specified intermediate position, or a stop (the scene interrupts any current movement).

6.4.5.9 Output restore values

As mentioned in an earlier paragraph, the status of the device after some significant events (see "*Output restore"* paragraph for description) can be defined by configuration.

The values available for restore after system events for coupled inputs are:

- None
- Up / Open
- Down / Close
- Stop
- Move to position

Further details can be found in the configuration section.

6.5 Device settings

This section lists all configurable parameters and describes related communication objects.

IMPORTANT:
All throughout this manual, the listed numbers for Communication Objects are respective to the 8/16-fold output module EK-FF1-TP.
For the 4/8-fold output module EK-FE1-TP, all CO numbers must be diminished by 1.

Every channel offers the same set of communication objects and parameters, but they may all be independently configured.

Hereafter, a generic channel number is referenced as “x” (where x = 1…8).

The parameter values highlighted in bold represent the default value.

Parameter name	*Conditions*	*Settings*
Manual operation	-	**enabled** disabled
	Enables the front panel pushbutton that activates manual mode.	
Disable from bus	Manual operation = enabled	**yes** no
	Allows to disable manual mode through a bus command	
Restore auto mode time	Manual operation = enabled Disable from bus = yes	hh:mm:ss (**00:15:00**)
	Sets the time after which the manual operation mode is reverted to automatic	
Device power off alarm	-	enabled **disabled**
	Makes an alarm communication objects available which signals when the auxiliary power supply fails.	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Disable front pushbuttons	Manual operation = enabled Disable from bus = yes	1 bit	C-W--	[1.002] boolean	1
Device power off alarm	Device power off alarm = enabled	1 bit	CR-T-	[1.005] alarm	2

The remaining device settings are divided in two main groups: the general channel configuration settings and the channel-specific settings.

6.5.1 Channels configuration

These settings configure which channels of the device are activated and in which mode.

Activating a channel causes the creation of a few communication objects in the minimal number required to switch the output relays through a bus telegram.

For outputs 2 and above, instead of being explicitly defined, the channel configuration can be copied from any of the preceding channels. If this option is selected, the corresponding channel can be made to perform the exact same kind of function as the source channel.

This allows to spare time in configuring the device, at the same time assuring that there is no inconsistency between two channels that are meant to be configured in exactly the same way.

It must be noted that to copy the configuration from another channel is just a shortcut for the selection of configuration options; it is in no way implied that the two channels share any of the involved communication objects. If the configuration of the original channel is varied, then so is the "derived" channel; in the same fashion, if the original channel is disabled, so is also the derived one.

Parameter name	*Conditions*	*Settings*
Channel *x*	-	disabled **2 binary outputs** valve / venetian blind / shutter copy parameters from channel*
	** This option is only available for channels nr. 2 and above.*	
Channel *x* – Source channel	Channel x = copy parameters from channel	**1**…*(x-1)*
Output *x*A	Channel x = 2 binary outputs	disabled **enabled**
	Enable first output of channel x.	
Output *x*B	Channel x = 2 binary outputs	disabled enabled **copy parameters from output xA**
	Enable second output of channel x.	
Channel *x* – Use	Channel x = valve / venetian blind / shutter	**valve** shutter venetian blind
	Type of configuration for the input pair	
Channel *x* – Three-way mode	Channel x = valve / venetian blind / shutter **Use = valve**	disabled / **enabled**
	Configures a valve for three-way mode (same functionality as for a shutter)	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output xA [xB] – On/off Command	Channel *x* = 2 binary outputs	1 bit	CRWTU	[1.001] on/off	3, 22, 58, 77, 113, 132, 168, 187, 223, 242, 278, 297, 333, 352, 388, 407
	This communication object is the standard "handle" for switching the output through a bus command.				
Channel *x* – Move up-down command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.008] up/down [1.009] open/close	41, 96, 151, 206, 261, 316, 371, 426
	Trigger object for continuous movement: when received, it starts continuous movement in the specified direction.				
Channel *x* – Stop-step up-down command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.007] step	42, 97, 152, 207, 262, 317, 372, 427
	Trigger object for step movement: when received, and the actuator is at rest, it starts a step movement in the specified direction. If the actuator is not at rest, just stops current movement.				
Channel *x* – Dedicated Stop command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.017] trigger	43, 98, 153, 208, 263, 318, 373, 428
	Stop any ongoing movement when received.				
Channel *x* – Info move	Channel x = valve / venetian blind / shutter	1 bit	CR-T-	[1.008] up/down	44, 99, 154, 209, 264, 319, 374, 429
	Allows to query the current movement direction				
Channel *x* – Valid current abs position	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	CR-T-	[1.002] boolean	45, 100, 155, 210, 265, 320, 375, 430
	Signals that the actuator has reached the requested absolute position. *Issued on absolute position movement commands.*				
Channel *x* – Abs [valve / shutter / blind] position command	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	C-W--	[5.001] percentage (0..100%)	53, 108, 163, 218, 273, 328, 383, 438
	Sets the target absolute position to reach and starts actuator movement *For the venetian blinds, the position refers to the blinds' panel.*				

Object name	Conditions	Size	Flags	DPT	CO number(s)
Channel *x* – Abs [valve / shutter / blind] position status	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	CR-T-	[5.001] percentage (0..100%)	54, 109, 164, 219, 274, 329, 384, 439
	Yields the current absolute position of the actuator. *The position is computed from the sequence of requested movements and realigned whenever an endpoint is reached.* *For the venetian blinds, the position refers to the blinds' panel.*				
Channel *x* – Abs slats position command	Channel x = valve / venetian blind / shutter **Use = venetian blind**	1 bit	C-W--	[5.001] percentage (0..100%)	56, 110, 166, 220, 276, 330, 386, 440
	Sets the target absolute position for the slats to reach and starts actuator movement.				
Channel *x* – Abs slats position status	Channel x = valve / venetian blind / shutter **Use = venetian blind**	1 bit	CR-T-	[5.001] percentage (0..100%)	57, 111, 167, 221, 277, 331, 387, 441
	Yields the current absolute position of the slats. *The position is computed from the sequence of requested movements and realigned whenever an endpoint of the slats' rotation is reached.*				

6.5.2 Independent outputs: Output *xA* / *xB* configuration

This section lists all settings for the output channels when used as independent outputs.

6.5.2.1 Main parameters

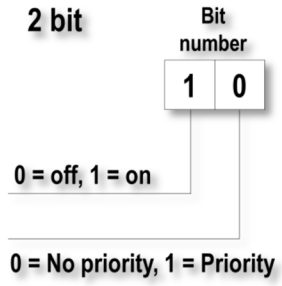
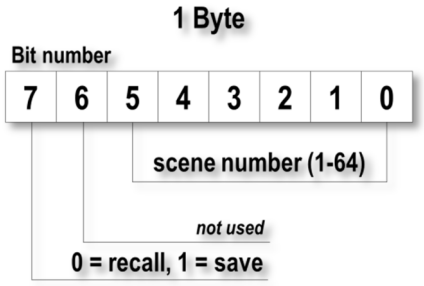
In this section most of the configurable parameters for the output are listed.

Parameter name	Conditions	Settings
Relay operation	Channel x = 2 binary outputs	**not inverted** inverted
	In the "not inverted" mode, the relay contacts (i.e. the physical output terminals) are shorted when the output is On (active).	
Behaviour at device power on	Channel x = 2 binary outputs	off on **no change** previous value
	Allows to determine the state of the output when the auxiliary power is restored.	
Behaviour at bus off	Channel x = 2 binary outputs	off on **no change**
	Allows to determine the state of the output when a bus voltage failure is detected	

Parameter name	*Conditions*	*Settings*
Behaviour at bus on	Channel x = 2 binary outputs	off on **no change** previous state
	Allows to determine the state of the output after bus recovery.	
Behaviour after download	Channel x = 2 binary outputs	off on **no change**
	Allows to determine the state of the output when the device resumes operation after a new parametrization has been downloaded.	
Status feedback telegram	Channel x = 2 binary outputs	disabled / **enabled**
	Enables or disables the output change notification through a bus telegram. *Updating the object from "ON" to "ON" or from "OFF" to "OFF" has no influence on the switching status feedback.*	
Status feedback telegram – Delay after bus voltage recovery	Channel x = 2 binary outputs **Status feedback telegram = enabled**	hh:mm:ss.fff (**00:00:03.000**)
	Time after bus voltage recovery before status feedback telegrams begin to be sent. *The delay has no effect on the behaviour of the outputs; only the feedback telegrams are delayed. The outputs can therefore be activated during the delay after a bus voltage recovery.* *During this delay, no feedback telegram will be transmitted even if a switching occurs; the feedback telegram for a switch during the delay period is lost.*	
Status feedback telegram – Transmission cycle time	Channel x = 2 binary outputs **Status feedback telegram = enabled**	hh:mm:ss (**00:00:00**)
	Interval between cyclical transmissions. *A zero value (00:00:00) means no cyclical transmission (feedback telegrams are only sent on value change).* *Values less than "00:00:10" (ten seconds) are considered by the firmware in any case as 10 (ten) seconds; the maximum value is 18:12:15.*	
On delay time	Channel x = 2 binary outputs	hh:mm:ss.fff (**00:00:00.000**)
	Delay between the "On" command telegram and the actual output activation. *This time delay does not affect the output of the staircase and forced control functions.* *For the scene function the delay can be set separately.* *Updating the object from "ON" to "ON" or from "OFF" to "OFF" retrigger the delay time.*	
Off delay time	Channel x = 2 binary outputs	hh:mm:ss.fff (**00:00:00.000**)
	Delay between the "Off" command telegram and the actual output deactivation. *Same comments as for the "On delay time" parameter apply.*	

Parameter name	*Conditions*	*Settings*
Staircase lighting function	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the staircase lighting feature. *For further details and parameter descriptions see the corresponding section below.*	
Locking function	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the capability of locking the input through a remote command. *For further details and parameter descriptions see the corresponding section below.*	
Forcing function	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the capability of forcing the input through a remote command. *For further details and parameter descriptions see the corresponding section below.*	
Forcing function - Behaviour end forced control	Channel x = 2 binary outputs **Forcing function = enabled**	off on **no change** previous value
	Allows to determine the state of the output when the forcing is released.	
Forcing function - Behaviour after bus recovery	Channel x = 2 binary outputs **Forcing function = enabled**	off on **no change** previous value
	Allows to determine the state of the output when the device resumes operation after bus voltage recovery. *Please notice that this is the status of the output, not the forcing status: forcing is maintained over bus failure and bus recovery.*	
Logic function	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the Logic input conditioning feature. *For further details and parameter descriptions see the corresponding section below.*	
Scenes function	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the Scene function. *For further details and parameter descriptions see the corresponding section below.*	
Operating energy / time counter	Channel x = 2 binary outputs	enabled / **disabled**
	Enables or disables the Hour / Energy counter function. *For further details and parameter descriptions see the corresponding section below.*	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – On/off status	Channel x = 2 binary outputs **Status feedback telegram = enabled**	1 bit	CR-T-	[1.001] switch	4, 23, 59, 78, 114, 133, 169, 188, 224, 243, 279, 298, 334, 353, 389, 408
	Sent at any change of the output state and also periodically, as configured.				
Output *xA [xB]* – Staircase lighting start stop command	Channel x = 2 binary outputs **Staircase lighting function = enabled**	1 bit	C-W--	[1.001] on/off	5, 24, 60, 79, 115, 134, 170, 189, 225, 244, 280, 299, 335, 354, 390, 409
	Starts the staircase light timing with an On value. *The timed activation automatically stops at the end of the preset time.* *If "Manual off" is enabled, the communication object will stop the timing with an Off value.*				
Output *xA [xB]* – Lock command	Channel x = 2 binary outputs **Locking function = enabled**	1 bit	C-W--	[1.003] enable	6, 25, 61, 80, 116, 135, 171, 190, 226, 245, 281, 300, 336, 355, 391, 410
	Inhibits the switching commands for the output when an "enable" telegram is received, and unlocks them when a "disable" telegram is received.				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – Forcing command	Channel x = 2 binary outputs **Forcing function = enabled**	2 bit	C-W--	[2.001] switch control	7, 26, 62, 81, 117, 136, 172, 191, 227, 246, 282, 301, 337, 356, 392, 411
	Allows to force the status of an output. *It is composed of 2 bits: the first one is used for the priority value (i.e. defines whether the forcing is in effect, "Priority", or not) and the second one for the imposed value (which is not considered if forcing is not effective).* 				
Output *xA [xB]* – Scene number	Channel x = 2 binary outputs **Scene function = enabled**	1 Byte	C-W--	[17.001] scene number [18.001] scene control	16, 35, 71, 90, 126, 145, 181, 200, 236, 255, 291, 310, 346, 365, 401, 420
	Allows to recall a scene setting for the status of the output, and to store current status in association to the specified scene. 				
Output *xA [xB]* – kWh counter	Channel x = 2 binary outputs **Operating hours / energy counter = enabled**	4-byte signed counter	CR-T-	[13.013] active energy [kWh]	17, 36, 72, 91, 127, 146, 182, 201, 237, 256, 292, 311, 347, 366, 402, 421
	Stores the current counter value of the accumulated energy.				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – kWh counter reset command	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	C-W--	[1.015] reset	18, 37, 73, 92, 128, 147, 183, 202, 238, 257, 293, 312, 348, 367, 403, 422
	Resets the energy counter to 0.				
Output *xA [xB]* – Hours counter	Channel x = 2 binary outputs Operating energy / time counter = enabled	2-byte unsigned counter	CR-T-	[7.007] time [h]	19, 38, 74, 93, 129, 148, 184, 203, 239, 258, 294, 313, 349, 368, 404, 423
	Stores the current counter value of the accumulated operating time.				
Output *xA [xB]* – Hours counter reset command	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	C-W--	[1.015] reset	20, 39, 75, 94, 130, 149, 185, 204, 240, 259, 295, 314, 350, 369, 405, 424
	Resets the operating hour counter to 0.				
Output *xA [xB]* – Hours counter runout	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	CR-T-	[1.005] alarm	21, 40, 76, 95, 131, 150, 186, 205, 241, 260, 296, 315, 351, 370, 406, 425
	1-bit alarm sent when the time counter reaches the maximum value of 65535 hours.				

6.5.2.2 Staircase lighting function

Parameter name	*Conditions*	*Settings*
Staircase lighting time	Channel x = 2 binary outputs Staircase lighting function = enabled	hh:mm:ss (**00:01:00**)
	Duration of staircase lighting time. *This time is the one shown on the time diagram in the descriptive section of this manual as "**Ts**".*	
Manual off	Channel x = 2 binary outputs Staircase lighting function = enabled	enabled / **disabled**
	When enabled, it allows an "Off" command to terminate the lighting time. *The "Off" command can be sent at any time with the same effect, including when the pre-warning is activated.*	
Retriggerable	Channel x = 2 binary outputs Staircase lighting function = enabled	enabled / **disabled**
	When enabled, it allows a new "On" command to restart the timing. *The "On" command can be sent at any time with the same effect, including when the pre-warning is activated.*	
Pre-warning	Channel x = 2 binary outputs Staircase lighting function = enabled	enabled / **disabled**
	Activates the pre-warning feature. *For a detailed description see the corresponding section of this manual.*	
Pre-warning – Pre-warning time	Channel x = 2 binary outputs Staircase lighting function = enabled **Pre-warning = enabled**	hh:mm:ss (**00:00:10**)
	Specifies how much time before the end of the timing a pre-warning light interruption will be carried out. *The time interval specified includes the interruption time.* *The maximum value is 18:12:15.* *This time is the one shown on the time diagram in the descriptive section of this manual as "**Tp-w**".*	
Pre-warning – Interruption time	Channel x = 2 binary outputs Staircase lighting function = enabled **Pre-warning = enabled**	hh:mm:ss.fff (**00:00:00.500**)
	Specifies the duration of the pre-warning interruption. *This time is the one shown on the time diagram in the descriptive section of this manual as "**Ti**".*	

- The pre-warning time should be shorter than the staircase time ($T_{P-W} < T_S$) and the interruption time shorter than the pre-warning time ($T_I < T_{P-W}$).
- Time delays have no influence on the staircase function (if enabled).
- A staircase timing in progress will be terminated by a reset of the actuator (bus voltage recovery or ETS reprogramming) or by using any function that affects the output (i.e. normal switching, forced control, logic function, scene recall), even if the function does not cause an actual change in the output value.
  On a forced termination, the value of the output remains unchanged; the same that is true also if the termination occurs during pre-warning time.

6.5.2.3 Locking function

Parameter name	*Conditions*	*Settings*
Lock device signal	Channel x = 2 binary outputs Locking function = enabled	**not inverted** / inverted
	Allows to interpret a "lock activate" telegram as unlock and vice-versa.	
After bus recovery	Channel x = 2 binary outputs Locking function = enabled	unlock lock **previous state**
	Defines how to set the lock status after bus voltage recovery.	
Behaviour at locking	Channel x = 2 binary outputs Locking function = enabled	**off** on no change
	Defines how to set the output value when the lock is activated.	
Behaviour at unlocking	Channel x = 2 binary outputs Locking function = enabled	**off** on no change updated value value before locking
	Defines how to set the output value when the lock is deactivated. ***Updated value** is the latest one that the output would assume if it had not been locked, i.e. it includes the output value change generated by whatever other function in the meantime.* ***Value before locking** is the value that the output had before the lock was activated.*	

6.5.2.4 Logic function

Parameter name	*Conditions*	*Settings*
Logic operation type	Channel x = 2 binary outputs Logic function = enabled	**OR** AND XOR
	Defines the logic operation to perform on allowable inputs.	
Read delay after bus recovery	Channel x = 2 binary outputs Logic function = enabled	hh:mm:ss.fff (**00:00:10.000**)
	After a bus voltage recovery, the device waits for the specified time before validating the logic objects used as inputs; a request is sent for each logical object value which has not arrived within the read delay. *The maximum value is 00:10:55.350.*	
Logic object *n*	Channel x = 2 binary outputs Logic function = enabled	**disabled** / enabled
	Defines which logic object is used as input. *Disabled logic objects are completely ignored and corresponding communication objects do not appear.*	
Logic object *n* – Logic object *n* negated	Channel x = 2 binary outputs Logic function = enabled **Logic object *n* = enabled**	**no** / yes
	Applies a logical negation to the value of the input object.	

The logic function is carried out only if and when at least one of the enabled input objects is updated by a bus telegram.

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – Logic Object *n*	Channel x = 2 binary outputs Logic function = enabled **Logic object *n* = enabled**	1 bit	CRWTU	[1.*] *generic 1-bit*	*Out 1A*: 8…15 *Out 1B*: 27…34 *Out 2A*: 63…70 *Out 2B*: 82…89 *Out 3A*: 118…125 *Out 3B*: 137…144 *Out 4A*: 173…180 *Out 4B*: 192…199 *Out 5A*: 228…235 *Out 5B*: 247…254 *Out 6A*: 283…290 *Out 6B*: 302…309 *Out 7A*: 338…345 *Out 7B*: 357…364 *Out 8A*: 393…400 *Out 8B*: 412…419
	For each output, the CO numbers corresponding to logic objects 1 to 8 are listed.				

6.5.2.5 Scenes function

Parameter name	*Conditions*	*Settings*
Download overwrites learned behavior	Channel x = 2 binary outputs Scenes function = enabled	no / **yes**
	Defines whether the download of a program on the device should erase and overwrite the stored scene output values previously learned and stored in the device. *When the device is put into operation for the first time, this parameter should be set to "yes" (default value) so that the output is initialized with valid scene values. Otherwise, the values are set to "0" (off) for all scenes.*	
Scene *n*	Channel x = 2 binary outputs Scenes function = enabled	enabled / **disabled**
	Enables or disables a new scene code to be assigned to the output.	
Scene *n* – Scene number	Channel x = 2 binary outputs Scenes function = enabled **Scene *n* = enabled**	1…64 (**1**)
	Scene number to be assigned to the output. The output will respond to scene commands that match the specified number.	
Scene *n* – Output behavior	Channel x = 2 binary outputs Scenes function = enabled **Scene *n* = enabled**	off / **on**
	(Initial) output value for the selected scene. This value will be possibly overwritten by a scene "store" command if the "Learning mode" option is enabled.	
Scene *n* – Activation delay	Channel x = 2 binary outputs Scenes function = enabled **Scene *n* = enabled**	hh:mm:ss.ff (**00:00:00.00**)
	Delay between a scene "recall" command and the actual output switching. *The maximum value is 01:49:13.50.*	
Scene *n* – Learning mode	Channel x = 2 binary outputs Scenes function = enabled **Scene *n* = enabled**	**disabled** / enabled
	When disabled, the scene "store" commands are ignored and only the output values set in the configuration are used.	

- Each scene recall telegram restarts the activation delay.
- If a new scene recall telegram is received while a delay is active (scene recall not yet executed), the old - and not yet recalled - scene will be rejected and the newest scene value will be in effect.
- The scene recall delay has no influence on the saving of scene values when the learning mode is active.
- If the same scene number is set for several scene entries, only the scene with the lowest entry number (1...8) will be considered. The other internal scenes will be ignored in this case.
- The scene recall can be overridden by a *forced control* or a *lock* function.

6.5.2.6 Watts / Hours counter

Parameter name	*Conditions*	*Settings*
Output load [W]	Channel x = 2 binary outputs Operating hours / energy counter = enabled	-671088640…+670760960 (**1000**)
	Defines the nominal rated power to be considered in computing the accumulated power consumption for the load connected to this output. *The total energy consumed [kWh] is calculated as the product of the specified value [W] and the operating hours [h].*	
Consumption / hours cyclic sending	Channel x = 2 binary outputs Operating hours / energy counter = enabled	hh:mm:ss (**00:00:00**)
	Defines the time interval for the cyclic retransmission of the counter values (both for accumulated time and energy). *A value of zero (00:00:00) disables cyclic transmission.*	

- During ETS programming or bus voltage failure, the counter stops counting.

6.5.3 Coupled outputs: Channel *x* configuration

This section lists all detail settings for the output channels when used as coupled outputs.

For all entries in this section, the condition "Channel x = valve / venetian blind / shutter" is implied, but not indicated for the sake of clarity.

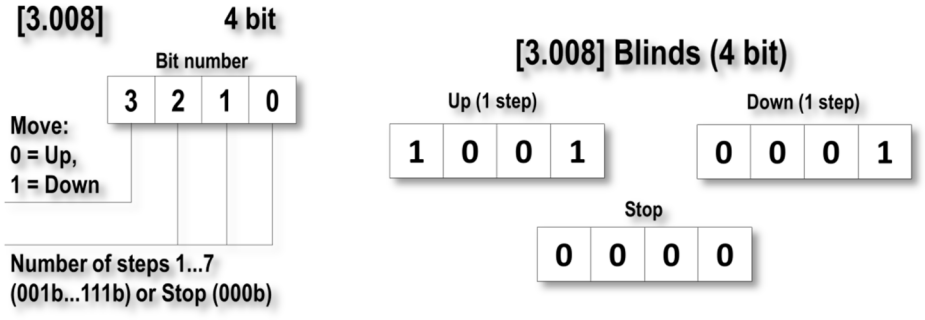
6.5.3.1 Main parameters

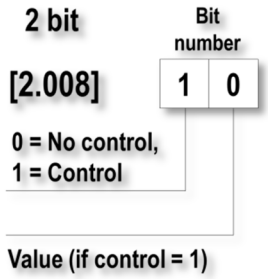
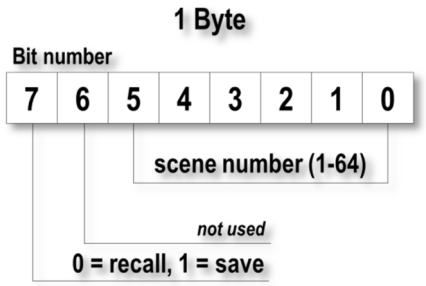
In this section most of the configurable parameters for the output are listed.

Parameter name	*Conditions*	*Settings*
Reversion pause time	***Use = all except 2-way valve***	0…65535 [Milliseconds] (**300 ms**)
	The minimum pause time between contact activation when switching from one output to another.	
Open time	***Use = all except 2-way valve***	hh:mm:ss (**00:00:15**)
	The time for the actuator to run the full stroke between the endpoints, in the opening direction. *It is important that the specification of this time is particularly accurate, since the accuracy of positioning depends heavily on it.*	
Close time	***Use = all except 2-way valve***	hh:mm:ss (**00:00:15**)
	The time for the actuator to run the full stroke between the endpoints, in the closing direction. *It is important that the specification of this time is particularly accurate, since the accuracy of positioning depends heavily on it.*	
Position control with dimmer	***Use = all except 2-way valve***	**no** / yes
	If this option is selected, a dimmer-type communication object is made available for the control of the actuator. It can be used, as an alternative, at the same time as the other standard control mechanisms.	

Parameter name	*Conditions*	*Settings*
Slat movement time	***Use = venetian blind***	hh:mm:ss (**00:00:15**)
	The time for the actuator to run the slats over the full stroke between the endpoints. *Unlike the main panel movement, there are no separate times for the two directions, because no relevant mechanical asymmetry is to be expected.* *It is important that the specification of this time is particularly accurate, since the accuracy of positioning depends heavily on it.*	
Slat step time	***Use = venetian blind***	0…65535 [Milliseconds] (**100 ms**)
	The activation time corresponding to a desired step span for the slats..	
Slats control with dimmer	***Use = venetian blind***	**no** / yes
	If this option is selected, a dimmer-type communication object is made available for the control of the actuator. It can be used, as an alternative, at the same time as the other standard control mechanisms.	
Behaviour at device power on	-	**none** up / open down / close stop move to position
	Allows to determine the state of the output when the auxiliary power is restored, provided the bus power supply had not failed (so the device has remained online).	
Behaviour at bus off	-	**none** up / open down / close stop move to position
	Allows to determine the state of the output when a bus voltage failure is detected.	
Behaviour at bus on	-	**none** up / open down / close stop move to position
	Allows to determine the state of the output after bus recovery.	
Behaviour after download	-	**none** up / open down / close stop move to position
	Allows to determine the state of the output when the device resumes operation after a new parametrization has been downloaded.	
Locking function	-	enabled / **disabled**
	Enables or disables the capability of locking the input through a remote command. *For further details and parameter descriptions see the corresponding section below.*	

Parameter name	*Conditions*	*Settings*
Forcing function	-	enabled / **disabled**
	Enables or disables the capability of forcing the input through a remote command. *For further details and parameter descriptions see the corresponding section below.*	
Forcing function - Behaviour end forced control	Forcing function = enabled	**none** up / open down / close stop move to position previous
	Allows to determine the state of the output when the forcing is released.	
Forcing function - Behaviour after bus recovery	Forcing function = enabled	**not forced** forced up / open forced down / closed stop move to position previous
	Allows to determine the state of the output when the device resumes operation after bus voltage recovery.	
Meteo alarms	-	enabled / **disabled**
	Enables or disables the Meteo alarm processing feature. *For further details and parameter descriptions see the corresponding section below.*	
Scenes function	-	enabled / **disabled**
	Enables or disables the Scene function. *For further details and parameter descriptions see the corresponding section below.*	

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Channel *x* – Dimmer blind position command	***Use = all except 2-way valve*** **Position control** **with dimmer = yes**	3-bit controlled	C-W--	[3.008] blind control	52, 107, 162, 217, 272, 327, 382, 437
	Allows to command the actuator through a dimmer-style command. 				
Channel *x* – Dimmer slats command	**Use = venetian blind** **Slats control** **with dimmer = yes**	3-bit controlled	C-W--	[3.008] blind control	55, 110, 165, 220, 275, 330, 385, 440
	Allows to command the slats position through a dimmer-style command. *See previous entry for bit field details.*				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Channel *x* – Lock command	**Locking function = enabled**	1 bit	C-W--	[1.003] enable	46, 101, 156, 211, 266, 321, 376, 431
	Inhibits the switching commands for the output when an "enable" telegram is received, and unlocks them when a "disable" telegram is received.				
Channel *x* – Forcing command	**Forcing function = enabled**	2 bit	C-W--	[2.008] direction 1 control	47, 102, 157, 212, 267, 322, 377, 432
	Allows to force the status of an output pair. The command is a "direction control" telegram, which can force movement in one direction, the other, or release forcing. 				
Channel *x* – Wind alarm	**Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	48, 103, 158, 213, 268, 323, 378, 433
	If this alarm is enabled, writing an active alarm value here will set the corresponding alarm condition; the alarm will be released by writing a "clear alarm" value. *If the heartbeat timeout" is set, even in absence of an alarm condition, the "clear alarm" value must be regularly written at intervals not higher than the timeout period.*				
Channel *x* – Frost alarm	**Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	49, 104, 159, 214, 269, 324, 379, 434
	Same considerations as for previous alarm apply.				
Channel *x* – Rain alarm	**Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	50, 105, 160, 215, 270, 325, 380, 435
	Same considerations as for previous alarm apply.				
Channel *x* – Scene number	**Scene function = enabled**	1 Byte	C-W--	[17.001] scene number [18.001] scene control	51, 106, 161, 216, 271, 326, 381, 436
	Allows to recall a scene setting for the status of the output, and to store current status in association to the specified scene. 				

6.5.3.2 Locking function

Parameter name	*Conditions*	*Settings*
Lock device signal	Locking function = enabled	**not inverted** / inverted
	Allows to interpret a "lock activate" telegram as unlock and vice-versa.	
After bus recovery	Locking function = enabled	unlock lock **previous state**
	Defines how to set the lock status after bus voltage recovery.	
Behaviour at locking	Locking function = enabled	**none** up / open down / close stop move to position
	Defines how to set the output value when the lock is activated.	
Behaviour at unlocking	Locking function = enabled	**none** up / open down / close stop move to position previous
	Defines how to set the output value when the lock is deactivated.	

6.5.3.3 Meteo Alarms

Parameter name	*Conditions*	*Settings*
Reaction to wind / frost / rain	Meteo Alarms = enabled	**none** up / open down / close
	Defines the position to be reached by the actuator when the alarm is active.	
Wind / frost / rain heartbeat timeout	Meteo Alarms = enabled	0…65535 [Minutes] (**10 Min.**)
	Defines the timeout for the alarm heartbeat. *If a heartbeat timeout is set, the alarm telegrams are required to be sent at regular intervals (shorter that the specified timeout), even with the alarm is not active, in order to be sure that the alarm communication is effective. If a "no alarm" telegram is not received in time, the alarm condition is set.* *A timeout value of zero (0) disables the heartbeat monitoring function.*	

Parameter name	*Conditions*	*Settings*
End of alarm action	Meteo Alarms = enabled	**none** up / open down / close previous
	Defines the position to be reached by the actuator when the alarm ceases.	

6.5.3.4 Scenes function

Parameter name	*Conditions*	*Settings*
Download overwrites learned behavior	Scenes function = enabled	no / **yes**
	Defines whether the download of a program on the device should erase and overwrite the stored scene output values previously learned and stored in the device. *When the device is put into operation for the first time, this parameter should be set to "yes" (default value) so that the output is initialized with valid scene values. Otherwise, the values are set to "0" (off) for all scenes.*	
Scene *n*	Scenes function = enabled	enabled / **disabled**
	Enables or disables a new scene code to be assigned to the output.	
Scene *n* – Scene number	Scenes function = enabled **Scene *n* = enabled**	1…64 (**1**)
	Scene number to be assigned to the output. The output will respond to scene commands that match the specified number.	
Scene *n* – Output behavior	Scenes function = enabled **Scene *n* = enabled**	stop **fully opened** fully closed move to position
	(Initial) output value for the selected scene. *This value will be possibly later overwritten by a scene "store" command if the "Learning mode" option is enabled.*	
Scene *n* – Scene position	Scenes function = enabled **Scene *n* = enabled** **Output behavior = move to position**	*(cursor control 0…100%)*
	Absolute position value for the blinds for the selected scene. *This value will be possibly later overwritten by a scene "store" command if the "Learning mode" option is enabled.*	
Scene *n* – Scene slat position	Scenes function = enabled **Scene *n* = enabled** **Output behavior = move to position** **Use = venetian blind**	*(cursor control 0…100%)*
	Absolute position value for the slats for the selected scene. *This value will be possibly later overwritten by a scene "store" command if the "Learning mode" option is enabled.*	

Parameter name	*Conditions*	*Settings*
Scene *n* – Activation delay	Scenes function = enabled **Scene *n* = enabled**	hh:mm:ss.ff (**00:00:00.00**)
	Delay between a scene "recall" command and the actual output switching. *The maximum value is 01:49:13.50.*	
Scene *n* – Learning mode	Scenes function = enabled **Scene *n* = enabled**	**disabled** / enabled
	When disabled, the scene "store" commands are ignored and only the output values set in the configuration are used.	

- Each scene recall telegram restarts the activation delay.
- If a new scene recall telegram is received while a delay is active (scene recall not yet executed), the old - and not yet recalled - scene will be rejected and the newest scene value will be in effect.
- The scene recall delay has no influence on the saving of scene values when the learning mode is active.
- If the same scene number is set for several scene entries, only the scene with the lowest entry number (1...8) will be considered. The other internal scenes will be ignored in this case.
- The scene recall can be overridden by a *forced control* or a *lock* function.

7 Appendix

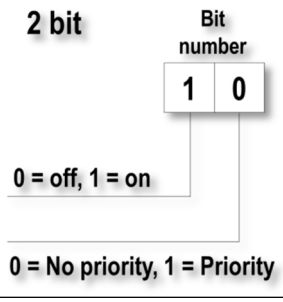
7.1 Communication objects table

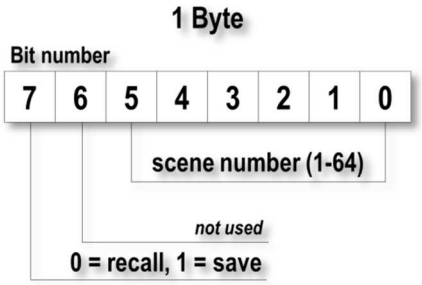
Following is a summary of all KNX Communication Objects (CO) and corresponding Data Point Types (DPT) defined by the application program according to configuration options.

IMPORTANT:
All throughout this manual, the listed numbers for Communication Objects are respective to the 8/16-fold output module EK-FF1-TP.
For the 4/8-fold output module EK-FE1-TP, all CO numbers must be diminished by 1.

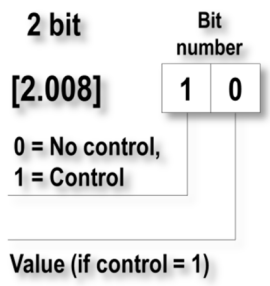
The listing order is generally by CO number.

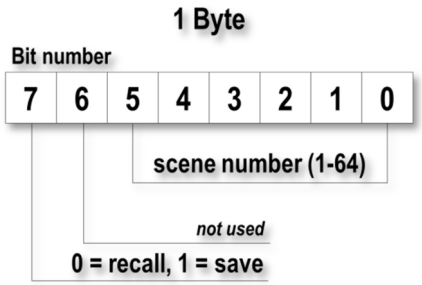
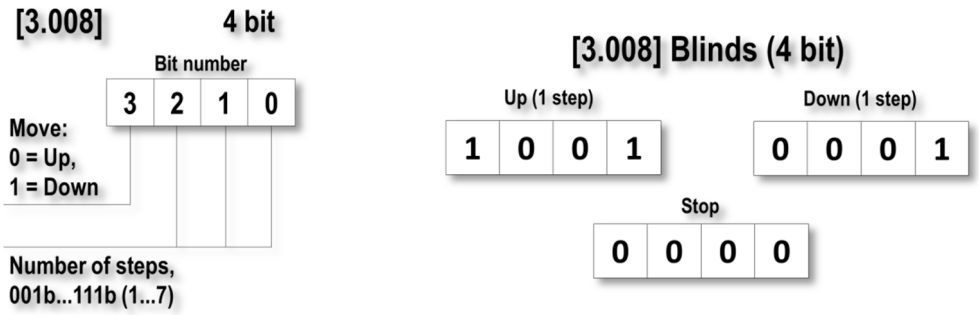
Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Disable front pushbuttons	Manual operation = enabled Disable from bus = yes	1 bit	C-W--	[1.002] boolean	1
Device power off alarm	Device power off alarm = enabled	1 bit	CR-T-	[1.005] alarm	2
Output xA [xB] – On/off Command	Channel *x* = 2 binary outputs	1 bit	CRWTU	[1.001] on/off	3, 22, 58, 77, 113, 132, 168, 187, 223, 242, 278, 297, 333, 352, 388, 407
	This communication object is the standard "handle" for switching the output through a bus command.				
Output *xA [xB]* – On/off status	Channel x = 2 binary outputs **Status feedback telegram = enabled**	1 bit	CR-T-	[1.001] switch	4, 23, 59, 78, 114, 133, 169, 188, 224, 243, 279, 298, 334, 353, 389, 408
	Sent at any change of the output state and also periodically, as configured.				

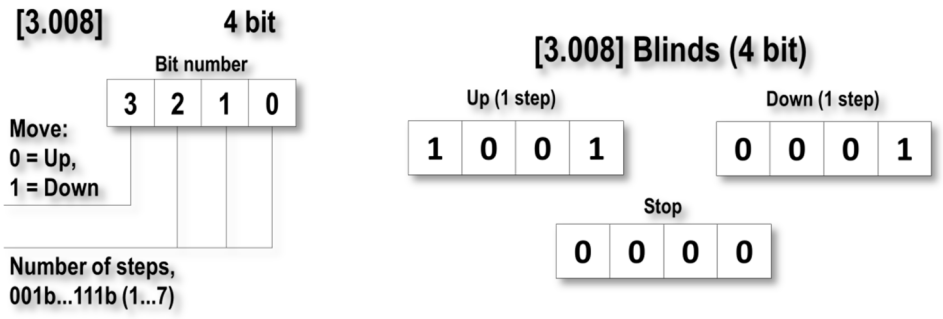
Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – Staircase lighting start stop command	Channel x = 2 binary outputs **Staircase lighting function = enabled**	1 bit	C-W--	[1.001] on/off	5, 24, 60, 79, 115, 134, 170, 189, 225, 244, 280, 299, 335, 354, 390, 409
	Starts the staircase light timing with an On value. *The timed activation automatically stops at the end of the preset time.* *If "Manual off" is enabled, the communication object will stop the timing with an Off value.*				
Output *xA [xB]* – Lock command	Channel x = 2 binary outputs **Locking function = enabled**	1 bit	C-W--	[1.003] enable	6, 25, 61, 80, 116, 135, 171, 190, 226, 245, 281, 300, 336, 355, 391, 410
	Inhibits the switching commands for the output when an "enable" telegram is received, and unlocks them when a "disable" telegram is received.				
Output *xA [xB]* – Forcing command	Channel x = 2 binary outputs **Forcing function = enabled**	2 bit	C-W--	[2.001] switch control	7, 26, 62, 81, 117, 136, 172, 191, 227, 246, 282, 301, 337, 356, 392, 411
	Allows to force the status of an output. *It is composed of 2 bits: the first one is used for the priority value (i.e. defines whether the forcing is in effect, "Priority", or not) and the second one for the imposed value (which is not considered if forcing is not effective).* 				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – Logic Object *n*	Channel x = 2 binary outputs Logic function = enabled **Logic object *n* = enabled**	1 bit	CRWTU	[1.*] *generic 1-bit*	*Out 1A*: 8…15 *Out 1B*: 27…34 *Out 2A*: 63…70 *Out 2B*: 82…89 *Out 3A*: 118…125 *Out 3B*: 137…144 *Out 4A*: 173…180 *Out 4B*: 192…199 *Out 5A*: 228…235 *Out 5B*: 247…254 *Out 6A*: 283…290 *Out 6B*: 302…309 *Out 7A*: 338…345 *Out 7B*: 357…364 *Out 8A*: 393…400 *Out 8B*: 412…419
	For each output, the CO numbers corresponding to logic objects 1 to 8 are listed.				
Output *xA [xB]* – Scene number	Channel x = 2 binary outputs **Scene function = enabled**	1 Byte	C-W--	[17.001] scene number [18.001] scene control	16, 35, 71, 90, 126, 145, 181, 200, 236, 255, 291, 310, 346, 365, 401, 420
	Allows to recall a scene setting for the status of the output, and to store current status in association to the specified scene.				
Output *xA [xB]* – kWh counter	Channel x = 2 binary outputs **Operating hours / energy counter = enabled**	4-byte signed counter	CR-T-	[13.013] active energy [kWh]	17, 36, 72, 91, 127, 146, 182, 201, 237, 256, 292, 311, 347, 366, 402, 421
	Stores the current counter value of the accumulated energy.				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Output *xA [xB]* – kWh counter reset command	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	C-W--	[1.015] reset	18, 37, 73, 92, 128, 147, 183, 202, 238, 257, 293, 312, 348, 367, 403, 422
	Resets the energy counter to 0.				
Output *xA [xB]* – Hours counter	Channel x = 2 binary outputs Operating energy / time counter = enabled	2-byte unsigned counter	CR-T-	[7.007] time [h]	19, 38, 74, 93, 129, 148, 184, 203, 239, 258, 294, 313, 349, 368, 404, 423
	Stores the current counter value of the accumulated operating time.				
Output *xA [xB]* – Hours counter reset command	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	C-W--	[1.015] reset	20, 39, 75, 94, 130, 149, 185, 204, 240, 259, 295, 314, 350, 369, 405, 424
	Resets the operating hour counter to 0.				
Output *xA [xB]* – Hours counter runout	Channel x = 2 binary outputs Operating energy / time counter = enabled	1 bit	CR-T-	[1.005] alarm	21, 40, 76, 95, 131, 150, 186, 205, 241, 260, 296, 315, 351, 370, 406, 425
	1-bit alarm sent when the time counter reaches the maximum value of 65535 hours.				
Channel *x* – Move up-down command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.008] up/down [1.009] open/close	41, 96, 151, 206, 261, 316, 371, 426
	Trigger object for continuous movement: when received, it starts continuous movement in the specified direction.				
Channel *x* – Stop-step up-down command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.007] step	42, 97, 152, 207, 262, 317, 372, 427
	Trigger object for step movement: when received, and the actuator is at rest, it starts a step movement in the specified direction. If the actuator is not at rest, just stops current movement.				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Channel *x* – Dedicated Stop command	Channel x = valve / venetian blind / shutter	1 bit	C-W--	[1.017] trigger	43, 98, 153, 208, 263, 318, 373, 428
	Stop any ongoing movement when received.				
Channel *x* – Info move	Channel x = valve / venetian blind / shutter	1 bit	CR-T-	[1.008] up/down	44, 99, 154, 209, 264, 319, 374, 429
	Allows to query the current movement direction				
Channel *x* – Valid current abs position	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	CR-T-	[1.002] boolean	45, 100, 155, 210, 265, 320, 375, 430
	Signals that the actuator has reached the requested absolute position. *Issued on absolute position movement commands.*				
Channel *x* – Lock command	Channel x = valve / venetian blind / shutter **Locking function = enabled**	1 bit	C-W--	[1.003] enable	46, 101, 156, 211, 266, 321, 376, 431
	Inhibits the switching commands for the output when an "enable" telegram is received, and unlocks them when a "disable" telegram is received.				
Channel *x* – Forcing command	Channel x = valve / venetian blind / shutter **Forcing function = enabled**	2 bit	C-W--	[2.008] direction 1 control	47, 102, 157, 212, 267, 322, 377, 432
	Allows to force the status of an output pair. The command is a "direction control" telegram, which can force movement in one direction, the other, or release forcing. 				
Channel *x* – Wind alarm	Channel x = valve / venetian blind / shutter **Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	48, 103, 158, 213, 268, 323, 378, 433
	If this alarm is enabled, writing an active alarm value here will set the corresponding alarm condition; the alarm will be released by writing a "clear alarm" value. *If the heartbeat timeout" is set, even in absence of an alarm condition, the "clear alarm" value must be regularly written at intervals not higher than the timeout period.*				
Channel *x* – Frost alarm	Channel x = valve / venetian blind / shutter **Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	49, 104, 159, 214, 269, 324, 379, 434
	Same considerations as for previous alarm apply.				
Channel *x* – Rain alarm	Channel x = valve / venetian blind / shutter **Meteo Alarms = enabled**	1 bit	C-W--	[1.005] alarm	50, 105, 160, 215, 270, 325, 380, 435
	Same considerations as for previous alarm apply.				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Channel *x* – Scene number	Channel x = valve / venetian blind / shutter **Scene function = enabled**	1 Byte	C-W--	[17.001] scene number [18.001] scene control	51, 106, 161, 216, 271, 326, 381, 436
	Allows to recall a scene setting for the status of the output, and to store current status in association to the specified scene. 				
Channel *x* – Dimmer blind position command	Channel x = valve / venetian blind / shutter ***Use = all except 2-way valve*** **Position control with dimmer = yes**	3-bit controlled	C-W--	[3.008] blind control	52, 107, 162, 217, 272, 327, 382, 437
	Allows to command the actuator through a dimmer-style command. 				
Channel *x* – Abs [valve / shutter / blind] position command	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	C-W--	[5.001] percentage (0..100%)	53, 108, 163, 218, 273, 328, 383, 438
	Sets the target absolute position to reach and starts actuator movement *For the venetian blinds, the position refers to the blinds' panel.*				
Channel *x* – Abs [valve / shutter / blind] position status	Channel x = valve / venetian blind / shutter **Use = *all except 2-way valve***	1 bit	CR-T-	[5.001] percentage (0..100%)	54, 109, 164, 219, 274, 329, 384, 439
	Yields the current absolute position of the actuator. *The position is computed from the sequence of requested movements and realigned whenever an endpoint is reached.* *For the venetian blinds, the position refers to the blinds' panel.*				

Object name	*Conditions*	*Size*	*Flags*	*DPT*	*CO number(s)*
Channel *x* – Dimmer slats command	Channel x = valve / venetian blind / shutter **Use = venetian blind Slats control with dimmer = yes**	3-bit controlled	C-W--	[3.008] blind control	55, 110, 165, 220, 275, 330, 385, 440
	Allows to command the slats position through a dimmer-style command. 				
Channel *x* – Abs slats position command	Channel x = valve / venetian blind / shutter **Use = venetian blind**	1 bit	C-W--	[5.001] percentage (0..100%)	56, 110, 166, 220, 276, 330, 386, 440
	Sets the target absolute position for the slats to reach and starts actuator movement.				
Channel *x* – Abs slats position status	Channel x = valve / venetian blind / shutter **Use = venetian blind**	1 bit	CR-T-	[5.001] percentage (0..100%)	57, 111, 167, 221, 277, 331, 387, 441
	Yields the current absolute position of the slats. *The position is computed from the sequence of requested movements and realigned whenever an endpoint of the slats' rotation is reached; this happens when the duration of an uninterrupted movement in a same direction is at least as high as the full stroke time specified as parameter.*				

7.2 Warning

- Installation, electrical connection, configuration and commissioning of the device can only be carried out by qualified personnel
- Opening the housing of the device causes the immediate end of the warranty period
- ekinex® KNX defective devices must be returned to the manufacturer at the following address: SBS S.p.A. Via Circonvallazione s / n, I-28010 Miasino (NO) Italy

7.3 Other information

- This application manual is aimed at installers, system integrators and planners
- For further information on the product, please contact the ekinex® technical support at the e-mail address: support@ekinex.com or visit the website www.ekinex.com
- ekinex® is a registered trademark of SBS S.p.A.
- KNX® and ETS® are registered trademarks of KNX Association cvba, Brussels