# KNX Bus Devices




KNX Association

Artman Training Center - Iran

# Table of Contents


***Introduction 1 ..... ۴***

***Data stored in a KNX device 2 ..... ۶***

***System profiles 3 ..... ۷***

**General 3.1 ..... ۷**

**Detailed description of the above features 3.2 ..... ۸**

Access control 3.2.1 ..... ۸

KNX Serial number 3.2.2 ..... ۸

Interface objects 3.2.3 ..... ۸

Memory size 3.2.4 ..... ۹

***Classical application functions 4 ..... ۹***

**Dimming with start/stop telegram 4.1 ..... ۹**

**Dimming with cyclical telegrams 4.2 ..... ۱۰**

**Application function: 'dimming actuator' 4.3 ..... ۱۱**

**Application function: drive control sensor 4.4 ..... ۱۲**

**Application function: drive control 4.5 ..... ۱۳**

***Drive control object structure 5 ..... ۱۴***


www.KNXTrain.com

In this chapter, the following abbreviations are used:

**PEI =** Physical External Interface

**BCU** = Bus coupling unit

**AM** = Application module

**TRC** = Transceiver

**SR =** Shift register

**DAC** = Digital-Analog Converter

www.KNXTrain.com

# Introduction ۱

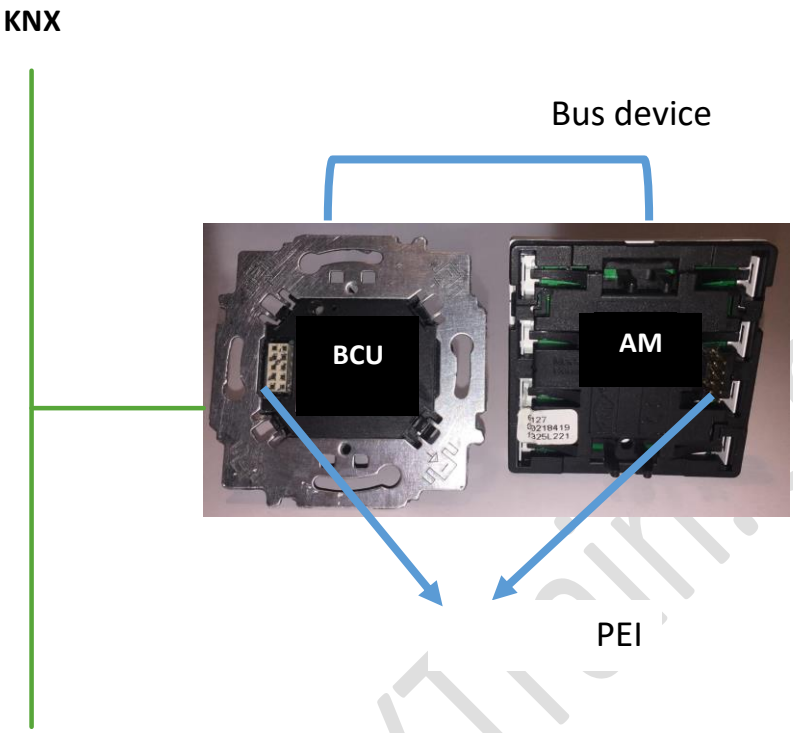



**Figure 1: Bus device consisting of a BCU and an application module**

A functioning bus device (e.g. dimming actuator, shutter actuator, a push button, a fire detection sensor…) principally consists of three interconnecting parts:

- bus coupling unit (BCU) ✓
- application module (AM) ✓
- application program (AP) ✓

In case of sensors, bus coupling units and application modules are offered on the market either physically separated or built into one housing. They must however always be sourced from the same manufacturer. When separated (sometimes and then mainly the case for flush-mounted sensors), the application module can be connected to the BCU via a standardized or a manufacturer-specific Physical External Interface (PEI). This PEI serves:

- as an interface to exchange messages between both parts ✓
- for providing limited power to the application module ✓

Whether the application module and bus coupling unit fit together – and whether they can be connected mechanically – has to be checked in the documentation as provided by the manufacturer.

In case of DIN rail-mounted devices, the bus coupling unit and application module are usually inseparably connected in one housing.

www.KNXTurk.com

**Note**: In the early days, a BCU could detect whether the application module stuck on it matched the loaded application program, by the measurement of a specific resistor of the application module over the PEI interface and comparing it to the PEI type of the loaded application program. In the case of a mismatch, the loaded application program could be stopped.
These days, manufacturers may opt for manufacturer specific mechanical coding to ensure that only the proper application module is stuck on (their) BCUs or manufacturers simply build integrated devices (where such a check is not needed).

In case of TP devices, the connection to the bus is mostly ensured via the standardized bus connector (dark grey/red).

Inside an integrated KNX product, from an electronics point of view, there is mostly:

- an electronic circuit (mostly chip) taking care of the coupling function to the respective medium (responsible for the sending and receiving of telegrams), called 'transceiver'; ✓
- a microcontroller running the system or operating software and application program and ✓
- an electronic circuit taking care of the application function. ✓

In general, the transceiver and microcontroller are integrated in a (regular) bus coupling unit of which the electronics cannot be separated. In some (rare) cases, the transceiver is offered as a separate module (e.g. a bus transceiver module).

Each bus device has its own intelligence thanks to the integrated operating system and program memory: This is the reason why KNX is a decentralized system and does not need a central supervising unit (e.g. a computer). Central functions (e.g. supervision) can however, if needed be realized via visualization and control software installed on PCs or tablet computers.

On the KNX product or the BCU, in most cases a programming button and a programming LED can be seen: the button is operated to put a BCU or the product in programming mode (= assignment of the individual address). The LED signals if the programming mode is on or off.

BCUs are currently available for connection to two different media: Twisted Pair (Safety Extra Low Voltage) and RF (KNX-RF).

Depending on their main function, bus devices can basically be divided into three classes: sensors, actuators and controllers. It is rare to have devices with pure sensor or actuator functionality nowadays. E.g., each push button with LED status display also has an "actuator" function and each actuator with status information has a "sensor" function.

- One possible function of the KNX sensor is e.g. to detect physical variables such as temperature, voltage, current… The application module of the sensor thus transfers information about its actual inputs (digital / analog) to the BCU. The BCU codes this data and sends it on the bus. The BCU therefore regularly checks the state of the inputs of the application module. ✓
- In the case of an actuator, the BCU receives telegrams from the bus, decodes them and passes this information on to the application module, which then will execute the command such as switching an output, e.g. an electric load (digital or analog loads). ✓
- A controller regulates the interaction between sensors and actuators (e.g. logical module) and has no physical inputs and outputs. ✓

## Data stored in a KNX device ۲

Inside the memory of a KNX device, the following data is stored:

- ✓ **The system software:** over the years, KNX and its manufacturers have standardized different versions of the system software (also referred to "system stack", or "KNX operating system"). These different software profiles are identified towards the ETS by means of a "mask version" or "device descriptor type 0". The mask version consists of 2 bytes and cannot be changed in the device. When downloading a device, the ETS checks this system software version to be able to make out how the device needs to be configured. Further information on these system profiles is given in clause ۳.
- ✓ **Temporary values of the system (RAM content) and the application:** These are lost when there is a bus power failure (if not saved earlier in non-volatile memory[1] by the device e.g. depending on the application program, important status information such as e.g. switching states).
- ✓ **the application program, the individual addresses, group addresses or parameters**: these are usually stored in rewritable memory.

In the case of S-mode compatible devices, the installer searches the matching ETS product data entry via the KNX Online Catalog or on the website of the manufacturer and then downloads it into the device, in this way determining the function of the product. An S-mode compatible KNX push button mounted on a BCU can only generate dimming signals, after the suitable application program has been programmed into the device via the ETS and the correct parameters have been set. The manufacturer code, which is different for each manufacturer, of the application program and the BCU must match in order to be able to download the application program.

If downloading an application program designed for a sensor into an actuator from the same manufacturer, two possible scenarios could occur:

۱. Usually, the BCU of the actuator will stop the download of the application program

۲. The application program is successfully downloaded into the BCU of the actuator without any error reported by ETS during the download process. The actuator will however not function.

In the case of E-mode devices, the device reports the supported functionality (as regards supported "E-mode" channels) by means of the device descriptor 2, indeed a different device descriptor than S-Mode devices. E-mode devices are normally shipped with loaded application program. The linking of such KNX devices and the setting of the relevant parameters is ensured either via appropriate hardware settings or via a central controller.

[1] Non-volatile memory has the capability to hold saved data even if the power is turned off.

# System profiles ۳

## General ۳٫۱

The System 1 technology was introduced with the first generation of KNX devices (in 1991), but is still part of the KNX Standard and can still be used in new devices.

A few years later in the nineties products based on System 2 and System 7 were introduced. System 7 was then further developed to System B in order to get rid of the limitations with regard to the number of group objects and group addresses.

The table below gives an overview of the most important features of these KNX system profiles:

| | System 1 | System 2/7 | System B |
|---|---|---|---|
| **Mask version or device descriptor type 0** | **E.g. 0012h** | **E.g. 0025h, 0705h** | **E.g. 07B0h** |
| **Maximum number of group objects[2]** | **12** | **254** | **65536** |
| **Support of interface objects** | **No** | **Yes** | |
| **Support KNX serial number** | | | |
| **Support access control (e.g. BCU key etc.)** | | | |

Application programs designed for System 1 technology can also be downloaded into certain System 2 devices (upwards compatibility). System 2 application programs can however not be downloaded into a System 1 device.

[2] The actual number of available group objects or group addresses which can be assigned depends on the used microcontroller

## Detailed description of the above features ۳٫۲

The features of System 2, System 7 and system B outlined above are explained here again in detail:

### Access control ۳٫۲٫۱

When a tool wants to access memory of System 2, 7 and B devices (reading and/or writing), it must first authorize itself by means of an authorization key of 4 bytes. Keys can be defined at various levels, some of the levels are reserved for manufacturer access (and can hence not be accessed by system integrators), one level however gives access to not system related memory in the devices and can be locked by entering a BCU key in the project properties. Devices that have been locked by means of a BCU key can still transmit telegrams. Only re-downloading a locked device without the proper BCU key, is not possible. BCU keys are stored in the ETS project. For this reason, it is vital to properly back-up ETS projects.

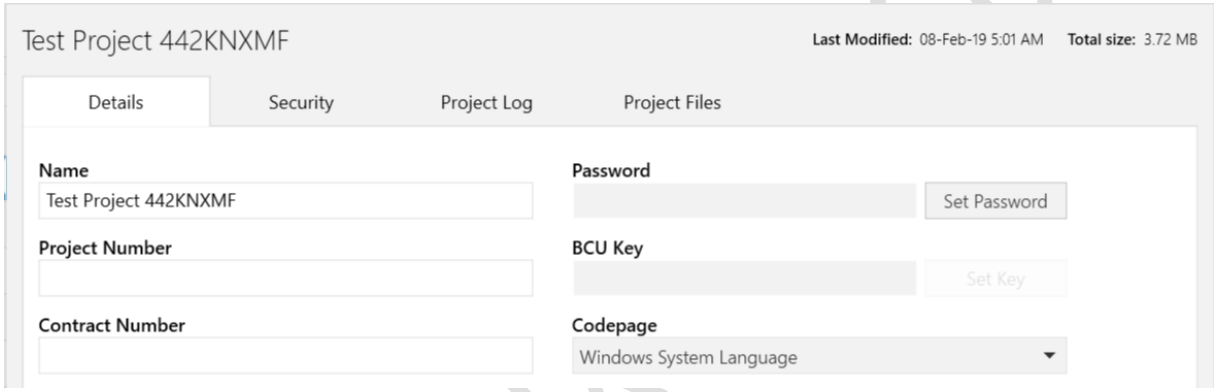



**Figure 2: entering a BCU key in the project details in ETS**

### KNX Serial number ۳٫۲٫۲

System 2, 7 and System B devices have a KNX serial number: this number, which is assigned to each device before leaving the factory, allows writing or reading the individual address of a device without having to press the programming button of the device. This feature is however not yet supported by default in ETS, but ETS Apps do exist in the MyKNX shop that allow to read/write the individual address without having to press the programming button.

### Interface objects ۳٫۲٫۳

Interface Objects contain certain system and application properties (e.g. address table, parameters …), which can be read and/or written by a tool (e.g. ETS during download) without explicit knowledge of the device's memory structure.

The ETS end user (= system integrator) cannot manipulate such interface objects but can read them by means of the ETS "Device Editor" App. More in-depth system knowledge explained during the KNX tutor course is required in order to do this kind of manipulation.

### Memory size ۳/۲/۴

When looking at the number of group objects and group addresses one can conclude that the memory size increases with the listed mask version: the memory size of a system 2 device is bigger than a system 1 device, the memory size of a system 7 device is bigger than a system 2 device, especially for profile “B”.

For more than 10 years, 254 group objects were considered as a large number but with the development of new touch panels, controllers and gateways, this number became too small again. That is why system 7 was extended with 1 byte as regards number of group addresses and group objects. Because of this, 65536 and 65535 have now become the maximum values, normally not reached by current applications. In other words, this is s a pure theoretical value, especially when one compares this to the total possible address capacity of a KNX system (which is just as big).

## Classical application functions ۴

### Dimming with start/stop telegram ۴/۱

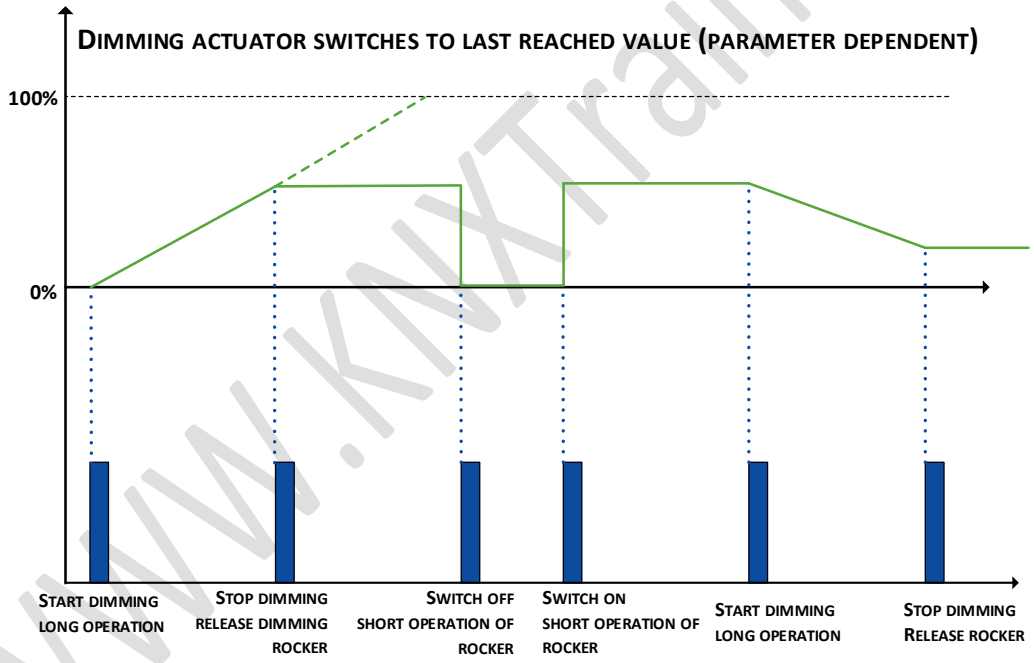



**Figure 3: Dimming with start/stop telegram**

The duration of the rocker operation determines whether the switching function or the dimming function via the same rocker is activated. If the time the rocker is pressed is shorter than the time parameterized in the application program of the push button (e.g. < 500 ms), a switch telegram is transmitted. If one operates the rocker longer than the time parameterized, a 'start dimming' telegram is transmitted. As soon as the rocker is released again, a 'stop dimming' telegram is transmitted.

Different group addresses are used for the switching and dimming telegrams to ensure that the dimming actuator executes the correct functions.

www.KNXTrain.com

## Dimming with cyclical telegrams ۴٫۲

**DIMMING SPEED OF THE ACTUATOR SHALL BE ADAPTED TO THE CYCLICAL TRANSMISSION OF DIMMING TELEGRAMS**

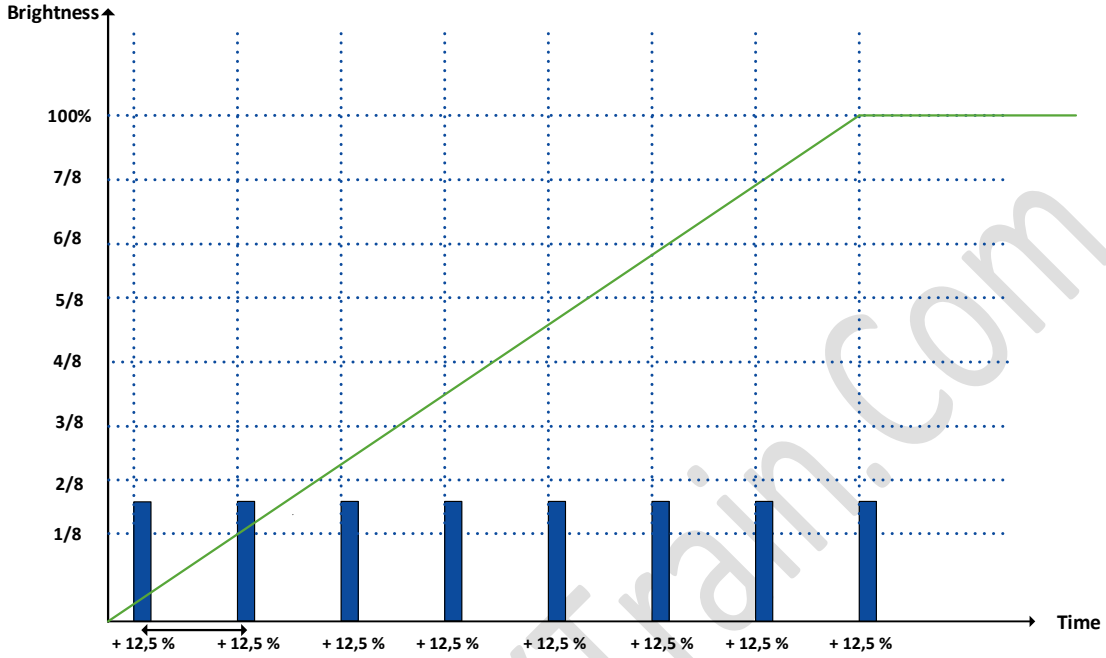



**Figure 4: Dimming with cyclical telegrams**

In a system controlled by wireless remote controls, e.g. infrared transmitter, the transmission signal might be interrupted as somebody passes through the IR beam. In order to avoid a situation where the dimming actuator does not receive important telegrams (e.g. the stop telegram), in most cases one will choose the setting 'cyclical dimming' during parameterization of a remote control. The transmitter in these settings transmits the telegram “increase brightness by 12.5 %”. The consequences of losing such a telegram are not as serious as the loss of a stop telegram, which is only sent once (as was explained in clause ۴٫۱).

www.KNXIran.com

## Application function: 'dimming actuator' ۴٫۳

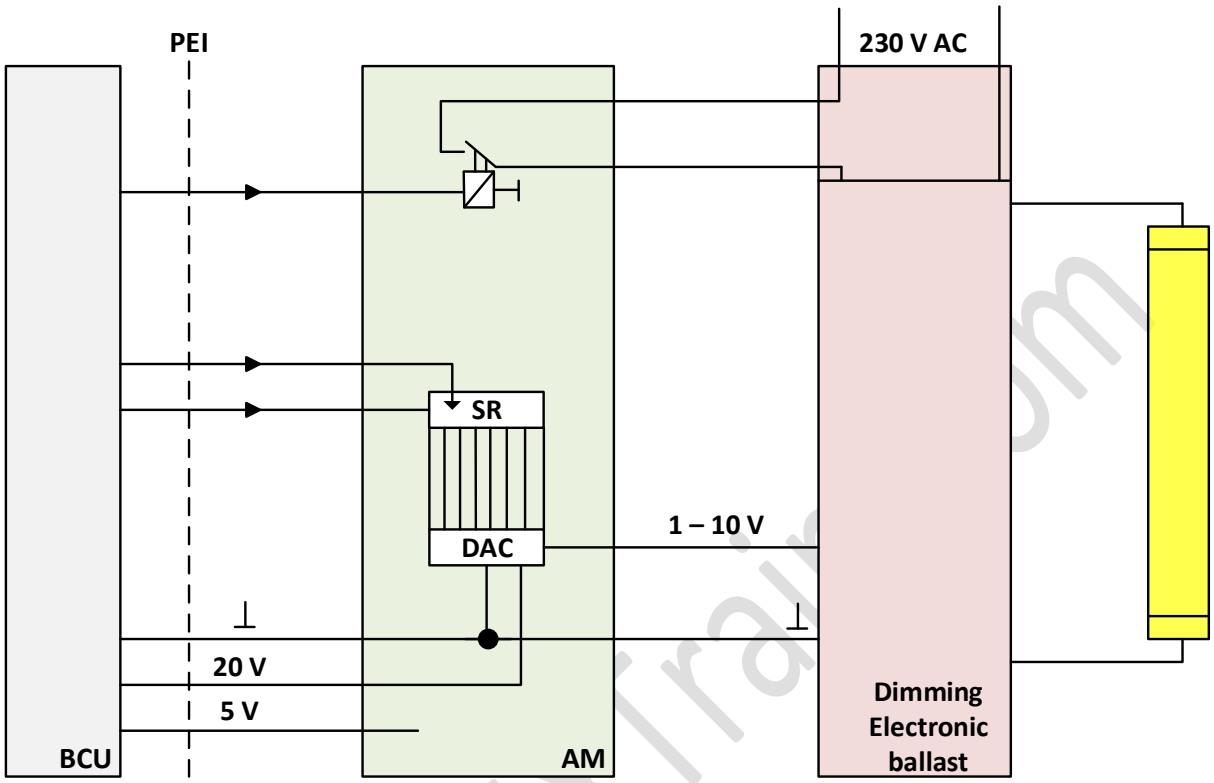



**Figure 5: Application function: 'dimming actuator'**

The counterpart to the sensor function dimming is the dimming actuator. There are various types of dimming actuators, depending on the dimming concept and the lamps or the ballasts used. In this example, a passive 1 – 10 V analog interface is shown. However, all dimming actuators have something in common: They have a parameterized dimming speed. The dimming speed is therefore an exclusive matter of the actuator.

In the example shown above, the BCU transmits a (digital) control signal to the application module. This signal has to be electronically adapted to the control input of the electronic ballast. The dimmer's electronic ballast uses the 1 – 10 voltage to control the brightness of a fluorescent tube. The relay in the application module is used to (dis)connect the mains voltage.

## Application function: drive control sensor ۴٫۴

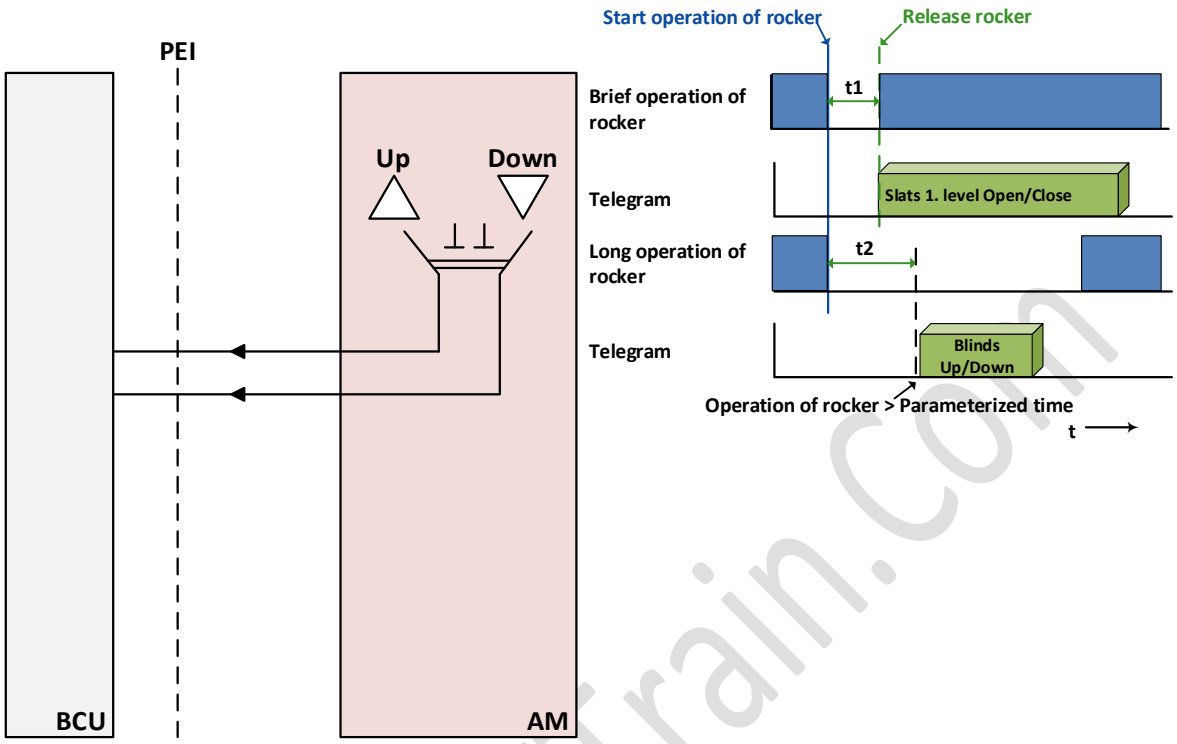



**Figure 6: Application function: drive control sensor**

The blinds, but also the shutter operation functions similarly to the dimming operation: A distinction is made between a brief and a long operation of the rocker.

The time t2 (e.g. 500 ms) acts as a "boundary" between the commands “slats step open/close” and “blinds up/down”.

T1 is the debouncing time that can be set for push button interfaces and binary inputs. For push buttons there is normally no debouncing time.

An important difference with dimming is however, that if one releases the rocker once the drive has started, the drive will continue to work until one again shortly presses the rocker.

This makes sense as blinds / shutters basically have much longer travel times compared to the time a dimming actuator needs for dimming up to 100 %.

The short operation of the rocker has also two different implications – when the drive is not in motion, it will cause a moving of the slats (only meaningful for blinds with adjustable slats). When sending the step command to a moving drive, this will cause the drive to stop.

This shows that in any case for blinds control, both commands i.e. shorter or longer operation of rocker are required, also when there is no need to adjust the slats.

www.KNXTrain.com

## Application function: drive control ۴٫۵

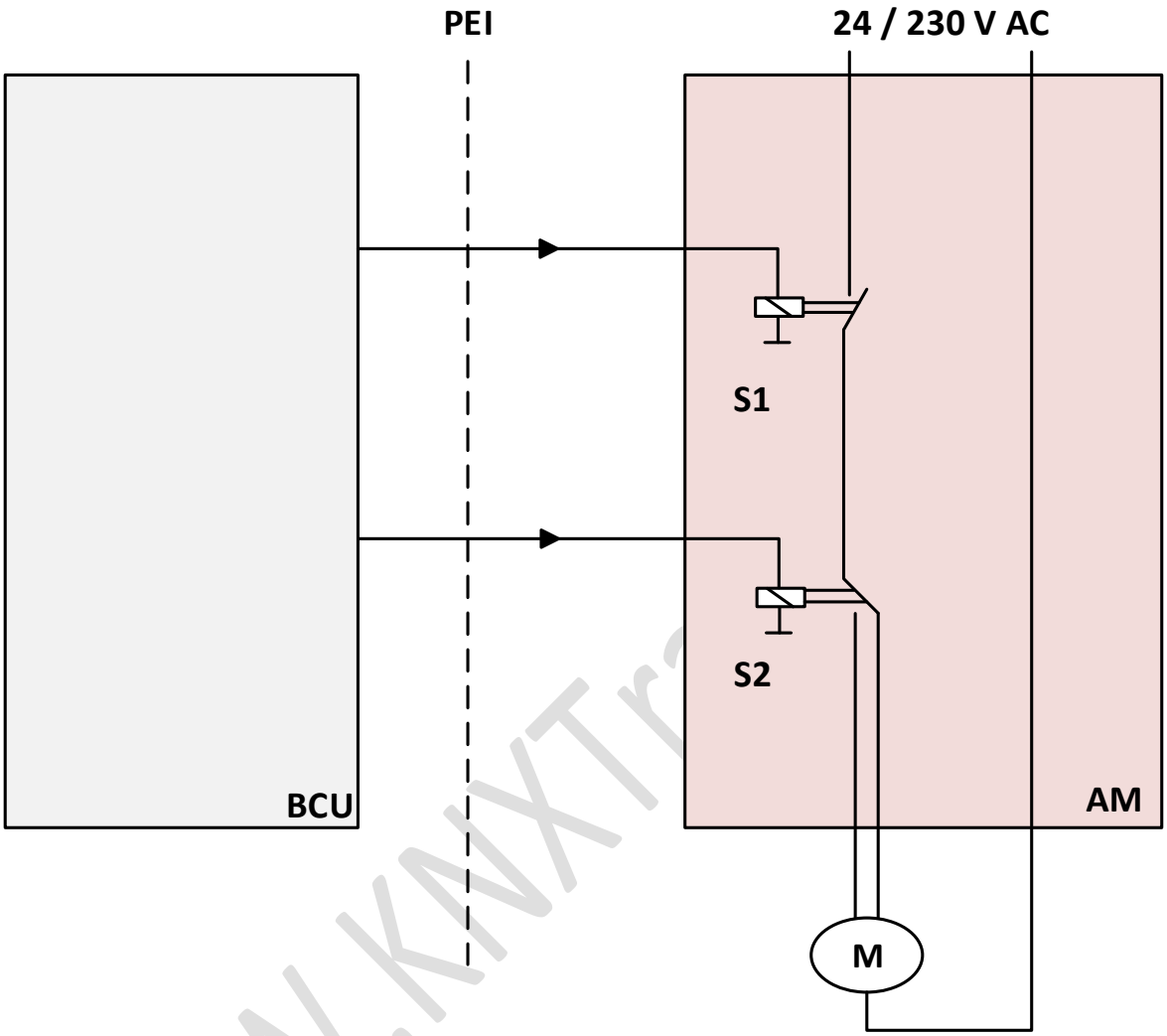



**Figure 7: Application function: drive control**

Depending on the telegram received, the BCU passes on the command “up” or the command “down” to the relay S2. On receiving the telegrams “slats open/close 1 step”', the BCU energizes the relay S1 for the appropriate duration. If the motor was already switched on, this telegram halts the blind (S1 opens). On receiving the telegram “blinds up/down”, the BCU energizes the relay S1 for a period longer than the total time the blind is in movement from the very top until the very bottom and vice versa. As usual, the limit switches of the blinds bring the motor to a halt when the limit position is reached, even if there is still voltage at the motor.

A blind actuator never has the task to take care of the safe deactivation of the blinds. This always needs to be ensured by the device itself in order to guarantee interlocking!

# Drive control object structure

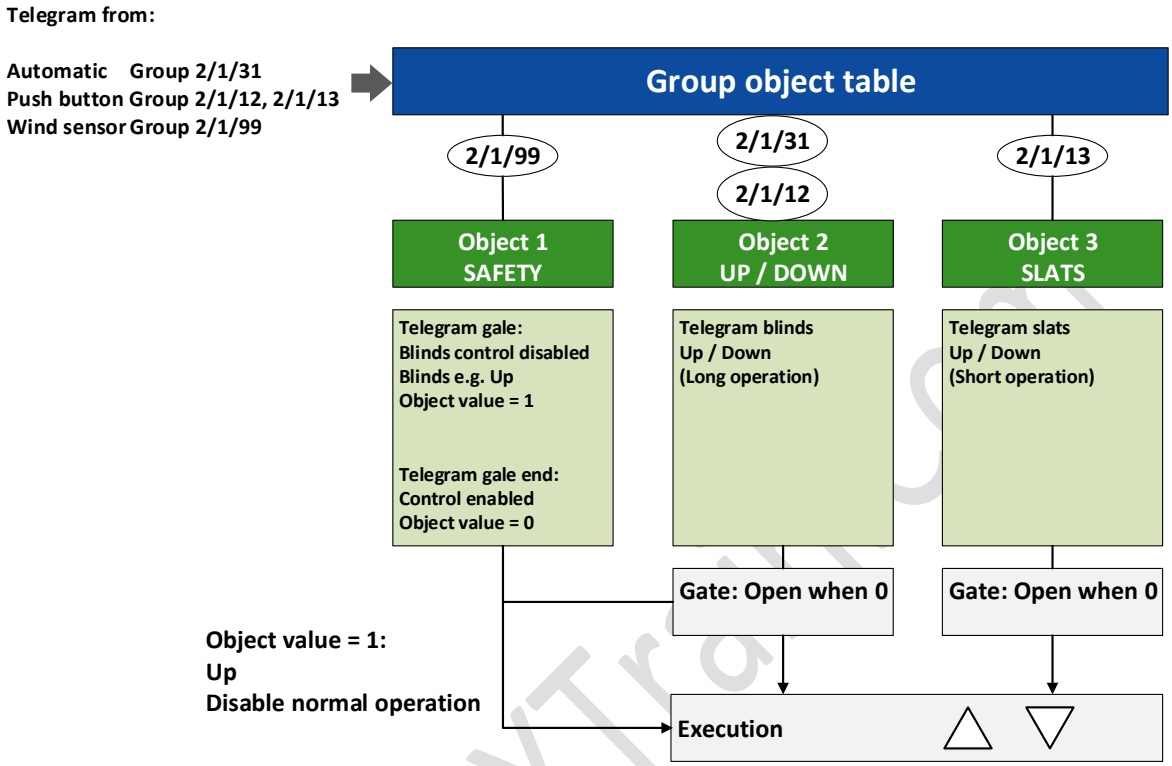



**Figure 8: Drive control object structure**

The figure above shows the basic functionality of a blind actuator. Apart from the normal operation, each blind/shutter actuator can for instance have a safety function.

If, for example, the sensor responsible for measuring the position of the sun triggers the telegram “blinds down” using the group address 2/1/31, the object group "up/down" is addressed and the corresponding command is executed.

Brief operation of the push button transmits the 2/1/13 telegram “adjust slats” and long operation of the key sensor sends the 2/1/12 telegram “open/close blinds completely”.

Telegram 2/1/99 triggered by the wind sensor addresses the group object “safety”. If a gale is developing, telegram 2/1/99 orders the blinds to go up/down (depending on the parameterization) and disables any further operation. When the storm has eased off, a telegram is sent that enables blind operation again. The de-activation of the alarm does not mean that the actuator is lowering the blinds again by itself (in the position before the gale). This makes no sense as the actuator has no information about the duration of the alarm or whether the blind really has to go down again.

New actuators have of course a variety of further functions and group objects, which cannot be explained during the basic course due to time constraints. These more complex functionalities, e.g. weather station, are explained in detail in the KNX advanced course.