Content

1 Introduction

1.1 Using the application program

Product family: Actuators
Product type: Actuators
Manufacturer: IPAS GmbH

Name: InBlock_i8HV
Order number: 77024-180-30

Product name	Order number
InBlock_i8HV	77024-180-30

1.2 General product information

1.2.1 General properties of the ETS application program

1.2.1.1 Installing the application program

The application for the InBlock_i8HV is based on a powerful KNX communications stack of the System-B type, with up to 1000 KNX objects. It is designed as a standard ETS application program and no plug-in for ETS-3 and ETS-4 is needed. After the import the product can be integrated as usual into the ETS. It can be found under product family “Input“ and product type “Actuators“.

1.2.2 Preliminary basic concepts

Input: Input type selection
In the InBlock_i8HV, each input is composed of two possibilities :

- Binary input
- Movement detector

Maximum sending speed
Should an output object be changed faster than the maximum sending speed of the KNX stack, these changes will be ignored and only the last change will be sent to the bus.

Cyclical sending
The application program contains multiple occasions where cyclic sending for different functions can be used. When this function is activated, the corresponding object will not send the telegram once, but repeat it infinitely.

Frequency and time calculation
The calculation of the preferred time (cyclical sending, delays, staircase, etc.) is done by multiplying the “time Base” by the “time Factor”.

Selection of data point type

During the configuration of the actuator, you will be asked to choose the data point type. It is very important to correctly define the DPT because this will change the size and type of the object; also, the data will be differently interpreted. E.g.: 1 Byte counter value = 0 to 255, whereas 1 Byte scaling value = 0 to 100%.

Additional/advanced functions (Function Block related)
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.
Also, see General_Settings_Advanced_Functions.

Scenes
In this actuator range we can find the Scenes controller (available in Advanced Functions): free configurable trigger conditions (start, save, stop and restore) and scene actions with time delays.

Enable/disable object
Most of the actuator's modules can be deactivated with a "… disable" object. The value (1 or 0) used to disable can also be configured.
This option can be very useful for many reasons, including simplifying the configuration: for instance, the logic functions might be a complex task that can take a while to finish; in the meantime, you don't want these modules to be active and cause unwanted actions. Therefore, you can disable them until you finish programming. Another example: you can simply activate/deactivate the timers for the irrigation system when not needed.

End-user parameters
It is very important for the end user to be able to change (via dedicated objects linked, for instance, to a visualization) certain settings of his/her KNX installation. This actuator allows for these changes to be maintained even when downloading the application program again. In "overwrite end-user parameter values at download" you will find an in-depth explanation on when and how to overwrite/maintain the changes made by the end-user.

2 ETS communication objects overview

The InBlock_i8HV device communicates via the KNX bus based on powerful communication stacks. Altogether 998 communication objects are available for the communication.

No.	Text	Function text	Object Size	Flags	Datapoint type
GENERAL OBJECTS					
	Central function block input	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
Each and every Function Block can individually be configured to have no reaction, switch ON / OFF or start the timer 1 reaction at on when this object receives a parametrized value. See parameter description to see all possibilities.					
	Central cyclic telegram for monitoring	> Cyclic ON telegrams	1 Bit	R-CT--	[[1.001] DPT_Switch
This object sends an ON telegram cyclic with bus voltage. This can be used to supervise a bus line. A channel in the mainline with a staircase timer can be triggered with a higher frequency than the staircase time by this object. Should the line fail the staircase will expire and therefor the "Line status light" will switch OFF.					
	Telegram at bus recovery	> Sends parameterized value	1 Bit	--CT--	[1.001] DPT_Switch
This object will send a parametrized value to the bus after bus voltage return. This can be used to trigger an event, like a scene to set up the whole installation at bus return.					
	Telegram at bus recovery	> Sends parameterized value	1 Byte	--CT--	[5.10] DPT_Value_1_Ucount
This object will send a parametrized value to the bus after bus voltage return. This can be used to trigger an event, like a scene to set up the whole installation at bus return.					
	Telegram at bus recovery	> Sends parameterized value	1 Byte	--CT--	[5.1] DPT_Scaling
This object will send a parametrized value to the bus after bus voltage return. This can be used to trigger an event, like a scene to set up the whole installation at bus return.					
	Telegram at bus recovery	> Sends parameterized value	2 Bytes	--CT--	[9] 9.xxx
This object will send a parametrized value to the bus after bus voltage return. This can be used to trigger an event, like a scene to set up the whole installation at bus return.					
	Manual control disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The manual buttons on the device can be deactivated by this object like this: Disable = 1 / Enable = 0					
	Manual control disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The manual buttons on the device can be deactivated by this object like this: Disable = 0 / Enable = 1					
ALARM OBJECTS					
	Alarm 1	< On / Off	1 Bit	RWC--I	[1.001] DPT_Switch

This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm 1	< 0..100%	1 Byte	RWC--I	[5.1] DPT_Scaling
This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm 1	< 1 byte unsigned	1 Byte	RWC--I	[5.10] DPT_Value_1_Ucount
This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm 1	< 2 bytes float	2 Bytes	RWC--I	[9] 9.xxx
This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm 1	< 4 bytes unsigned	4 Bytes	RWC--I	[12.1] DPT_Value_4_Ucount
This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm 1	< 4 bytes float	4 Bytes	RWC--I	[14] 14.xxx
This object is the alarm 1 trigger object. In the parameters one can define with which value it should be in the alarm state.					
	Alarm ACK	< Ack. with 0	1 Bit	-WC---	[1.016] DPT_Acknowledge
When activating the acknowledge function this object appears. This is to acknowledge the alarm by sending a 0 to this object. Alarms can only be acknowledged if the alarm has disappeared					
	Alarm ACK	< Ack. with 1	1 Bit	-WC---	[1.016] DPT_Acknowledge
When activating the acknowledge function this object appears. This is to acknowledge the alarm by sending a 1 to this object. Alarms can only be acknowledged if the alarm has disappeared					
	Alarm 1 setpoint	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
If the alarm is configured to be an analog alarm then the threshold of this alarm can be set by this object					
	Alarm 1 setpoint	< 0..100%	1 Byte	RWC---	[5.1] DPT_Scaling
If the alarm is configured to be an analog alarm then the threshold of this alarm can be set by this object					
	Alarm 1 setpoint	< 2 bytes float	2 Bytes	RWC---	[9] 9.xxx
If the alarm is configured to be an analog alarm then the threshold of this alarm can be set by this object					
	Alarm 1 setpoint	< 4 bytes unsigned	4 Bytes	RWC---	[12.1] DPT_Value_4_Ucount
If the alarm is configured to be an analog alarm then the threshold of this alarm can be set by this object					
	Alarm 1 setpoint	< 4 bytes float	4 Bytes	RWC---	[14] 14.xxx
If the alarm is configured to be an analog alarm then the threshold of this alarm can be set by this object					
	Alarm 1 hysteresis	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount

If the alarm is configured to be an analog alarm then the hysteresis of this alarm setpoint can be changed by this object					
	Alarm 1 hysteresis	< 0..100%	1 Byte	RWC---	[5.1] DPT_Scaling
If the alarm is configured to be an analog alarm then the hysteresis of this alarm setpoint can be changed by this object					
	Alarm 1 hysteresis	< 2 bytes float	2 Bytes	RWC---	[9] 9.xxx
If the alarm is configured to be an analog alarm then the hysteresis of this alarm setpoint can be changed by this object					
	Alarm 1 hysteresis	< 4 bytes float	4 Bytes	RWC---	[14] 14.xxx
If the alarm is configured to be an analog alarm then the hysteresis of this alarm setpoint can be changed by this object					
	Alarm 1 hysteresis	< 4 bytes unsigned	4 Bytes	RWC---	[12.1] DPT_Value_4_Ucount
If the alarm is configured to be an analog alarm then the hysteresis of this alarm setpoint can be changed by this object					
	Alarm 1 disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The alarm can be disabled by sending a 1 to this object.					
	Alarm 1 status	> ON = Alarm, OFF = No alarm	1 Bit	R-CT--	[1] 1.005 DPT_Alarm
This object will send the actual alarm status value					
LOGIC OBJECTS					
	Logic 1 disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The logic function can be disabled by sending a 0					
	Logic 1 disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The logic function can be disabled by sending a 1					
	Logic 1 input 1	< On / Off	1 Bit	RWCTU-	[1.001] DPT_Switch
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 0..100%	1 Byte	RWCTU-	[5.1] DPT_Scaling
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 1 byte signed	1 Byte	RWCTU-	[6.10] DPT_Value_1_Count
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 1 byte unsigned	1 Byte	RWCTU-	[5.10] DPT_Value_1_Ucount

This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 2 bytes unsigned	2 Bytes	RWCTU-	[7.1] DPT_Value_2_Ucount
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 2 bytes float	2 Bytes	RWCTU-	[9] 9.xxx
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 2 bytes signed	2 Bytes	RWCTU-	[8.1] DPT_Value_2_Count
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 4 bytes unsigned	4 Bytes	RWCTU-	[12.1] DPT_Value_4_Ucount
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 4 bytes float	4 Bytes	RWCTU-	[14] 14.xxx
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 1	< 4 bytes signed	4 Bytes	RWCTU-	[13.1] DPT_Value_4_Count
This is the first of 4 logic inputs of this logic block					
	Logic 1 input 2	< On / Off	1 Bit	RWCTU-	[1.001] DPT_Switch
This is the second of 4 logic inputs of this logic block					
	Logic 1 Enable / Disable Gate	< Disable = 1 / Enable = 0	1 Bit	RWCT--	[1.003] DPT_Enable
If the logic function is configured to be a Gate function, then this input is used to enable or disable the gate. When the gate is disabled the input will not be sent to the output. This object can also be used to trigger the input to the output with different conditions (please see the parameter description to see all possibilities)					
	Logic 1 Enable / Disable Gate	< Disable = 0 / Enable = 1	1 Bit	RWCT--	[1.003] DPT_Enable
If the logic function is configured to be a Gate function, then this input is used to enable or disable the gate. When the gate is disabled the input will not be sent to the output. This object can also be used to trigger the input to the output with different conditions (please see the parameter description to see al possibilities)					
	Logic 1 input 2	< 1 byte signed	1 Byte	RWCTU-	[6.10] DPT_Value_1_Count
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 0..100%	1 Byte	RWCTU-	[5.1] DPT_Scaling
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 1 byte unsigned	1 Byte	RWCTU-	[5.10] DPT_Value_1_Ucount
This is the second of 4 logic inputs of this logic block					

	Logic 1 input 2	< 2 bytes signed	2 Bytes	RWCTU-	[8.1] DPT_Value_2_Count
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 2 bytes unsigned	2 Bytes	RWCTU-	[7.1] DPT_Value_2_Ucount
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 2 bytes float	2 Bytes	RWCTU-	[9] 9.xxx
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 4 bytes unsigned	4 Bytes	RWCTU-	[12.1] DPT_Value_4_Ucount
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 4 bytes float	4 Bytes	RWCTU-	[14] 14.xxx
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 2	< 4 bytes signed	4 Bytes	RWCTU-	[13.1] DPT_Value_4_Count
This is the second of 4 logic inputs of this logic block					
	Logic 1 input 3	< On / Off	1 Bit	RWCTU-	[1.001] DPT_Switch
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 0..100%	1 Byte	RWCTU-	[5.1] DPT_Scaling
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 1 byte unsigned	1 Byte	RWCTU-	[5.10] DPT_Value_1_Ucount
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 1 byte signed	1 Byte	RWCTU-	[6.10] DPT_Value_1_Count
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 2 bytes unsigned	2 Bytes	RWCTU-	[7.1] DPT_Value_2_Ucount
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 2 bytes signed	2 Bytes	RWCTU-	[8.1] DPT_Value_2_Count
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 2 bytes float	2 Bytes	RWCTU-	[9] 9.xxx
This is the third of 4 logic inputs of this logic block					

	Logic 1 input 3	< 4 bytes unsigned	4 Bytes	RWCTU-	[12.1] DPT_Value_4_Ucount
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 4 bytes signed	4 Bytes	RWCTU-	[13.1] DPT_Value_4_Count
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 3	< 4 bytes float	4 Bytes	RWCTU-	[14] 14.xxx
This is the third of 4 logic inputs of this logic block					
	Logic 1 input 4	< On / Off	1 Bit	RWCTU-	[1.001] DPT_Switch
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 0..100%	1 Byte	RWCTU-	[5.1] DPT_Scaling
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 1 byte unsigned	1 Byte	RWCTU-	[5.10] DPT_Value_1_Ucount
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 1 byte signed	1 Byte	RWCTU-	[6.10] DPT_Value_1_Count
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 2 bytes unsigned	2 Bytes	RWCTU-	[7.1] DPT_Value_2_Ucount
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 2 bytes signed	2 Bytes	RWCTU-	[8.1] DPT_Value_2_Count
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 2 bytes float	2 Bytes	RWCTU-	[9] 9.xxx
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 4 bytes signed	4 Bytes	RWCTU-	[13.1] DPT_Value_4_Count
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 4 bytes float	4 Bytes	RWCTU-	[14] 14.xxx
This is the fourth of 4 logic inputs of this logic block					
	Logic 1 input 4	< 4 bytes unsigned	4 Bytes	RWCTU-	[12.1] DPT_Value_4_Ucount
This is the fourth of 4 logic inputs of this logic block					

	Logic 1 output	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 1 byte signed	1 Byte	R-CT--	[6.10] DPT_Value_1_Count
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
53	Logic 1 output	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 2 bytes signed	2 Bytes	R-CT--	[8.1] DPT_Value_2_Count
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 4 bytes signed	4 Bytes	R-CT--	[13.1] DPT_Value_4_Count
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
	Logic 1 output	> 4 bytes float	4 Bytes	R-CT--	[14] 14.xxx
This is the output of this logic block and the DPT can differ the input. The value when true or false or the result of the logic block will be sent with this object.					
SCENES OBJECTS					
	Scene 1 input	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 0..100%	1 Byte	-WC---	[5.1] DPT_Scaling
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					

	Scene 1 input	< 1 byte signed	1 Byte	-WC---	[6.10] DPT_Value_1_Count
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 1 byte unsigned	1 Byte	-WC---	[5.10] DPT_Value_1_Ucount
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 2 bytes unsigned	2 Bytes	-WC---	[7.1] DPT_Value_2_Ucount
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 2 bytes float	2 Bytes	-WC---	[9] 9.xxx
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 2 bytes signed	2 Bytes	-WC---	[8.1] DPT_Value_2_Count
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 4 bytes float	4 Bytes	-WC---	[14] 14.xxx
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 4 bytes signed	4 Bytes	-WC---	[13.1] DPT_Value_4_Count
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 input	< 4 bytes unsigned	4 Bytes	-WC---	[12.1] DPT_Value_4_Ucount
This is the input object to trigger a function of the scene. Different values for this function can be set in the parameters like the play, record, stop and restore values.					
	Scene 1 disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The scene can be disable with a 1					
	Scene 1 disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The scene can be disable with a 0					
	Scene 1 event 1	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the first event for the first scene.					
	Scene 1 event 1	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the first event for the first scene.					
	Scene 1 event 1	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the first event for the first scene.					

	Scene 1 event 1	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the first event for the first scene.					
	Scene 1 event 1	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the first event for the first scene.					
	Scene 1 event 1	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the first event for the first scene.					
	Scene 1 event 1	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the first event for the first scene.					
	Scene 1 event 1	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the first event for the first scene.					
	Scene 1 event 1	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the first event for the first scene.					
	Scene 1 event 1	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the first event for the first scene.					
	Scene 1 event 2	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the second event for the first scene.					
	Scene 1 event 2	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the second event for the first scene.					
	Scene 1 event 2	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the second event for the first scene.					
	Scene 1 event 2	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the second event for the first scene.					
	Scene 1 event 2	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the second event for the first scene.					
	Scene 1 event 2	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the second event for the first scene.					

	Scene 1 event 2	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the second event for the first scene.					
	Scene 1 event 2	<> 4 bytes un-signed	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the second event for the first scene.					
	Scene 1 event 2	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the second event for the first scene.					
	Scene 1 event 2	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the second event for the first scene.					
	Scene 1 event 3	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the third event for the first scene.					
	Scene 1 event 3	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the third event for the first scene.					
	Scene 1 event 3	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the third event for the first scene.					
	Scene 1 event 3	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the third event for the first scene.					
	Scene 1 event 3	<> 2 bytes un-signed	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the third event for the first scene.					
	Scene 1 event 3	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the third event for the first scene.					
	Scene 1 event 3	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the third event for the first scene.					
	Scene 1 event 3	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the third event for the first scene.					
	Scene 1 event 3	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the third event for the first scene.					

	Scene 1 event 3	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the third event for the first scene.					
	Scene 1 event 4	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the fourth event for the first scene.					
	Scene 1 event 4	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the fourth event for the first scene.					
	Scene 1 event 5	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the fifth event for the first scene.					

	Scene 1 event 5	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the fifth event for the first scene.					
	Scene 1 event 5	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the fifth event for the first scene.					
	Scene 1 event 6	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 1 byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the sixth event for the first scene.					

	Scene 1 event 6	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the sixth event for the first scene.					
	Scene 1 event 6	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the sixth event for the first scene.					
	Scene 1 event 7	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the seventh event for the first scene.					

	Scene 1 event 7	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the seventh event for the first scene.					
	Scene 1 event 7	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the seventh event for the first scene.					
	Scene 1 event 8	<> On / Off	1 Bit	-WCTU-	[1.001] DPT_Switch
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 1 byte signed	1 Byte	-WCTU-	[6.10] DPT_Value_1_Count
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 0..100%	1 Byte	-WCTU-	[5.1] DPT_Scaling
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 1byte unsigned	1 Byte	-WCTU-	[5.10] DPT_Value_1_Ucount
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 2 bytes unsigned	2 Bytes	-WCTU-	[7.1] DPT_Value_2_Ucount
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 2 bytes float	2 Bytes	-WCTU-	[9] 9.xxx
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 2 bytes signed	2 Bytes	-WCTU-	[8.1] DPT_Value_2_Count
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 4 bytes unsigned	4 Bytes	-WCTU-	[12.1] DPT_Value_4_Ucount
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 4 bytes signed	4 Bytes	-WCTU-	[13.1] DPT_Value_4_Count
This is the eighth event for the first scene.					
	Scene 1 event 8	<> 4 bytes float	4 Bytes	-WCTU-	[14] 14.xxx
This is the eighth event for the first scene.					
TIMERS OBJECTS					

	Timer 1 trigger	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is to trigger the first timer					
	Timer 1 trigger	< 1 byte signed	1 Byte	-WC---	[6.10] DPT_Value_1_Count
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 1 byte scaling	1 Byte	-WC---	[5.1] DPT_Scaling
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 1 byte unsigned	1 Byte	-WC---	[5.10] DPT_Value_1_Ucount
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 2 bytes unsigned	2 Bytes	-WC---	[7.1] DPT_Value_2_Ucount
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 2 bytes float	2 Bytes	-WC---	[9] 9.xxx
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 2 bytes signed	2 Bytes	-WC---	[8.1] DPT_Value_2_Count
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 4 bytes unsigned	4 Bytes	-WC---	[12.1] DPT_Value_4_Ucount
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 4 bytes signed	4 Bytes	-WC---	[13.1] DPT_Value_4_Count
This is to trigger the first timer (only for delay)					
	Timer 1 trigger	< 4 bytes float	4 Bytes	-WC---	[14] 14.xxx
This is to trigger the first timer (only for delay)					
	Timer 1 change factor/Remaining time	< 1 byte unsigned	1 Byte	RWCT--	[5.10] DPT_Value_1_Ucount
Change factor: With this object the ON time of the timer can be changed. If the base is equal to 1 second, this object will change the time in seconds. If the base is 1 minute the value sent to the object is equal to the minutes the staircase will be ON, etc. Remaining time: Additionally, to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value. In order to disable this function, the “T” flag must be deactivated.					
	Timer 1 warning pulse	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
An additional object can be activated to send a warning pulse to inform that the staircase is about to expire and therefore have time to react in order to trigger it again.					

	Timer 1 disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The timer can be disabled by this object by sending a 0					
	Timer 1 output	> On / Off	1 Bit	--CT--	[1.1] DPT_Switch
This is the output object of the timer.					
	Timer 1 output	> 1 byte signed	1 Byte	--CT--	[6.10] DPT_Value_1_Count
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 1 byte unsigned	1 Byte	--CT--	[5.10] DPT_Value_1_Ucount
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 1 byte scaling	1 Byte	--CT--	[5.1] DPT_Scaling
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 2 bytes float	2 Bytes	--CT--	[9] 9.xxx
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 2 bytes unsigned	2 Bytes	--CT--	[7.1] DPT_Value_2_Ucount
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 2 bytes signed	2 Bytes	--CT--	[8.1] DPT_Value_2_Count
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 4 bytes signed	4 Bytes	--CT--	[13.1] DPT_Value_4_Count
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 4 bytes unsigned	4 Bytes	--CT--	[12.1] DPT_Value_4_Ucount
This is the output object of the timer. (only for the delay function)					
	Timer 1 output	> 4 bytes float	4 Bytes	--CT--	[14] 14.xxx
This is the output object of the timer. (only for the delay function)					
SETPOINT OBJECTS					
	Setpoint 1 output value 1	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the output of the two-point regulator for the first setpoint. This output will switch ON or OFF depending on the parametrized values when crossing the threshold values					

	Setpoint 1 setpoint value/status	<> 0..100%	1 Byte	RWCT--	[5.1] DPT_Scaling
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 setpoint value/status	<> 1 byte unsigned	1 Byte	RWCT--	[5.10] DPT_Value_1_Ucount
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 setpoint value/status	<> 2 bytes float	2 Bytes	RWCT--	[9] 9.xxx
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 setpoint value/status	<> 2 bytes unsigned	2 Bytes	RWCT--	[7.1] DPT_Value_2_Ucount
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 setpoint value/status	<> 4 bytes float	4 Bytes	RWCT--	[14] 14.xxx
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 setpoint value/status	<> 4 bytes unsigned	4 Bytes	RWCT--	[12.1] DPT_Value_4_Ucount
The desired setpoint value can be adjusted with this object. The same object will be used to send the current setpoint status value. This status value will be sent when changing from heat to cool and depending on the parameters when blocking an unblocking the setpoint					
	Setpoint 1 Heat / Cool	< Heat = 1 / Cool = 0	1 Bit	RWC---	[1] 1.100
With this object the two-point regulator will change from heat to cool mode. This will cause the threshold to change from: (Lower threshold = Setpoint at Cool = 0) and (Upper threshold = Setpoint at Heat = 1)					
	Setpoint 1 input ext. sensor value	< 0..100%	1 Byte	RWC---	[5.1] DPT_Scaling
This is the analog value which will be used as the input for the setpoint					
	Setpoint 1 input ext. sensor value	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
This is the analog value which will be used as the input for the setpoint					
	Setpoint 1 input ext. sensor value	< 2 bytes float	2 Bytes	RWC---	[9] 9.xxx
This is the analog value which will be used as the input for the setpoint					
	Setpoint 1 input ext. sensor value	< 2 byte unsigned	2 Bytes	RWC---	[7.1] DPT_Value_2_Ucount
This is the analog value which will be used as the input for the setpoint					
	Setpoint 1 input ext. sensor value	< 4 bytes float	4 Bytes	RWC---	[14] 14.xxx
This is the analog value which will be used as the input for the setpoint					

	Setpoint 1 input ext. sensor value	< 4 bytes unsigned	4 Bytes	RWC---	[12.1] DPT_Value_4_Ucount
This is the analog value which will be used as the input for the setpoint					
	Setpoint 1 disable	< On / Off	1 Bit	RWC---	[1.003] DPT_Enable
The setpoint can be disabled with this object					
	Setpoint 1 disable	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
The setpoint can be disabled with this object. This can also be used to change the HVAC mode when linking this object of more than one setpoint to the same group address but with different enable values. E.g. If setpoint 1 is enabled by the value 1 and setpoint 2 by the value 2, then setpoint 1 can be the comfort mode and setpoint 2 standby mode.					
FUNCTION BLOCK OBJECTS					
	[A1] Function block input On / Off	< On / Off	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will receive a 1/ON or a 0/OFF value					
	[A1] Function block input toggle/inverted	< Inverted	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A1] Function block input toggle/inverted	< Toggle only with 0	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A1] Function block toggle/inverted	< Toggle with 0 and 1	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A1] Function block toggle/inverted	< Toggle only with 1	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A1] Function block output	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
This is the current output of the function block. The sending behaviour can be changed by the parameters					
	[A1] RunHour counter value	> 4 bytes signed	4 Bytes	R-CT--	[13.100] DPT_time_lag_(s)
The accumulated ON time of the function block is called the runhours and it is send by this object. The frequency and values to be sent can be changed in the application program. One can even apply different multiplying or division factors in the application.					
	[A1] RunHour counter threshold	< Reading/writing threshold	4 Bytes signed	RWCT--	[13.100] DPT_time_lag_(s)
The threshold of the runhour counter can be changed by this object. When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A1] RunHour counter threshold	< Reading threshold	4 Bytes	R-CT--	[13.100] DPT_time_lag_(s)

			signed		
The threshold of the runhour counter can be changed by this object. When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A1] RunHour counter alarm	> 1 = Alarm, 0 = No alarm	1 Bit	R-CT--	[1.005] DPT_Alarm
When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A1] RunHour counter reset	< 1 = Reset, 0 = Nothing	1 Bit	-WC---	[1.015] DPT_Reset
The runhour counter can be reset by this object in order to start counting again from zero. In the parameters one can decide to reset to zero or if the counter object should maintain and send the last value at reset					
	[A1] RunHour counter value at reset	> 4 bytes signed	4 Bytes	R-CT--	[13.100] DPT_time_lag_(s)
In the parameters one can decide to activate this object should store and send the last value of the runhour counter at reset.					
	[A1] Switching counter value	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A1] Switching counter value	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A1] Switching counter value	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A1] Switching counter threshold	< Reading/writing threshold	1 Byte	RWCT--	[5.10] DPT_Value_1_Ucount
This object is to read and write the threshold value.					
	[A1] Switching counter threshold	< Reading threshold	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This object is to only read the threshold value.					
	[A1] Switching counter threshold	< Reading threshold	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This object is to only read the threshold value.					
	[A1] Switching counter threshold	< Reading/writing threshold	2 Bytes	RWCT--	[7.1] DPT_Value_2_Ucount
This object is to read and write the threshold value.					
	[A1] Switching counter threshold	< Reading threshold	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This object is to only read the threshold value.					
	[A1] Switching counter threshold	< Reading/writing threshold	4 Bytes	RWCT--	[12.1] DPT_Value_4_Ucount
This object is to read and write the threshold value.					

	[A1] Switching counter alarm	> 1 = Alarm, 0 = No alarm	1 Bit	R-CT--	[1.005] DPT_Alarm
When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A1] Switching counter reset	< 1 = Reset, 0 = Nothing	1 Bit	-WC---	[1.015] DPT_Reset
The switching counter can be reset by this object in order to start counting again from zero. In the parameters one can decide to reset to zero or if the counter object should maintain and send the last value at reset					
	[A1] Switching counter value at reset	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A1] Switching counter value at reset	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A1] Switching counter value at reset	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A1] Scene number	< Sc1 (0=Play 128=Rec)... Sc64	1 Byte	-WC---	[5.10] DPT_Value_1_Ucount
With this object any of the configured scenes of this function block can be triggered and/or recorded.					
	[A1] Scene disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The scene function for this function block can be disabled by sending a 1 to this object					
	[A1] Scene disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The scene function for this function block can be disabled by sending a 0 to this object					
	[A1] Timer 1 trigger	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is to trigger the first timer associated to the function block					
	[A1] Timer 1 change factor/Remaining time	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
Change factor: With this object the ON time of the timer can be changed. If the base is equal to 1 second, this object will change the time in seconds. If the base is 1 minute the value sent to the object is equal to the minutes the staircase will be ON, etc. Remaining time: Additionally to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value. In order to disable this function, the “T” flag must be deactivated.					
	[A1] Timer 1 warning pulse	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
An additional object can be activated to send a warning pulse to inform that the staircase is about to expire and therefore have time to react in order to trigger it again.					
	[A1] Timer 1 disable	< Disable = 0 / Enable = 1	1 Bit	RWCT--	[1.003] DPT_Enable
With this object the timer will be disabled by receiving a 0					

	[A1] Timer 2 trigger	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is to trigger the second timer associated to the function block					
	[A1] Timer 2 change factor/Remaining time	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
Change factor: With this object the ON time of the timer can be changed. If the base is equal to 1 second, this object will change the time in seconds. If the base is 1 minute the value sent to the object is equal to the minutes the staircase will be ON, etc. Remaining time: Additionally to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value. In order to disable this function, the “T” flag must be deactivated.					
	[A1] Timer 2 warning pulse	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
An additional object can be activated to send a warning pulse to inform that the staircase is about to expire and therefore have time to react in order to trigger it again.					
	[A1] Timer 2 disable	< Disable = 0 / Enable = 1	1 Bit	RWCT--	[1.003] DPT_Enable
The timer can be disabled by this object by sending a 0					
	[A1] Disable function block	< On / Off	1 Bit	RWCT--	[1.003] DPT_Enable
The function block can be disabled by this object. In the parameters one can decide to disable with a 1 or a 0.					
	[A2] Function block input On / Off	< On / Off	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block will receive a 1/ON or an 0/OFF					
	[A2] Function block input toggle/inverted	< Toggle only with 1	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A2] Function block input toggle/inverted	< Toggle with 0 and 1	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A2] Function block input toggle/inverted	< Toggle only with 0	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A2] Function block input toggle/inverted	< Inverted	1 Bit	-WC---	[1.1] DPT_Switch
With this object the function block input will be inverted. But it can also be used to toggle the output regardless of the previous state of the output. The value to do this can also be configured in the parameters.					
	[A2] Function block output	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
This is the output of the function block. The sending behaviour can be changed by the parameters					
	[A] Scene number	< Sc1 (0=Play 128=Rec)... Sc64	1 Byte	-WC---	[5.10] DPT_Value_1_Ucount
With this object any of the configured scenes of this function block can be triggered and/or recorded.					

	[A] Scene disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
The scene function for this function block can be disabled by sending a 1 to this object					
	[A] Scene disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
The scene function for this function block can be disabled by sending a 1 to this object					
	[A2] RunHour counter value	> 4 bytes signed	4 Bytes	R-CT--	[13.100] DPT_time_lag_(s)
The runhour value of this function block will be sent to the bus. The frequency to be sent can be adjusted. It can also be set to send different values than hours, when using the advanced functions of the runhour. Please see the parameter description.					
	[A] Disable function block	< On / Off	1 Bit	RWCT--	[1.003] DPT_Enable
The function block can be disabled by this object. In the parameters one can decide to disable with a 1 or a 0.					
	[A2] RunHour counter threshold	< Reading threshold	4 Bytes signed	R-CT--	[13.100] DPT_time_lag_(s)
The threshold of the runhour counter can be changed by this object. When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A2] RunHour counter threshold	< Reading/writing threshold	4 Bytes signed	RWCT--	[13.100] DPT_time_lag_(s)
The threshold of the runhour counter can be changed by this object. When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A2] RunHour counter alarm	> 1 = Alarm, 0 = No alarm	1 Bit	R-CT--	[1.005] DPT_Alarm
When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A2] RunHour counter reset	< 1 = Reset, 0 = Nothing	1 Bit	-WC---	[1.015] DPT_Reset
The runhour counter can be reset by this object in order to start counting again from zero. In the parameters one can decide to reset to zero or if the counter object should maintain and send the last value at reset					
	[A2] RunHour counter value at reset	> 4 bytes signed	4 Bytes	R-CT--	[13.100] DPT_time_lag_(s)
In the parameters one can decide to activate this object and if it should store and send the last value of the runhour counter at reset.					
	[A2] Switching counter value	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A2] Switching counter value	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A2] Switching counter value	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This object sends the number of switching's, whether to count when in switches ON, OFF or both can be configured in the parameters					
	[A2] Switching counter threshold	< Reading threshold	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount

This object is to only read the threshold value.					
	[A2] Switching counter threshold	< Reading/writing threshold	1 Byte	RWCT--	[5.10] DPT_Value_1_Ucount
This object is to read and write the threshold value.					
	[A2] Switching counter threshold	< Reading/writing threshold	2 Bytes	RWCT--	[7.1] DPT_Value_2_Ucount
This object is to read and write the threshold value.					
	[A2] Switching counter threshold	< Reading thresh-old	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This object is to only read the threshold value.					
	[A2] Switching counter threshold	< Reading/writing threshold	4 Bytes	RWCT--	[12.1] DPT_Value_4_Ucount
This object is to read and write the threshold value.					
	[A2] Switching counter threshold	< Reading thresh-old	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This object is to only read the threshold value.					
	[A2] Switching counter alarm	> 1 = Alarm, 0 = No alarm	1 Bit	R-CT--	1.005] DPT_Alarm
When crossing the threshold value the threshold alarm object will send an alarm message.					
	[A2] Switching counter reset	< 1 = Reset, 0 = Nothing	1 Bit	-WC---	[1.015] DPT_Reset
The switching counter can be reset by this object in order to start counting again from zero. In the parame-ters one can decide to reset to zero or if the counter object should maintain and send the last value at reset					
	[A2] Switching counter value at reset	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A2] Switching counter value at reset	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A2] Switching counter value at reset	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
In the parameters one can decide to activate this object and if it should store and send the last value of the switching counter at reset.					
	[A2] Scene number	< Sc1 (0=Play 128=Rec)... Sc64	1 Byte	-WC---	[18.001] DPT_Scene_control
With this object any of the configured scenes of this function block can be triggered and/or recorded.					
	[A2] Scene disable	< Disable = 1 / En-able = 0	1 Bit	RWC---	[1.003] DPT_Enable
The scene function for this function block can be disabled by sending a 1 to this object					
	[A2] Scene disable	< Disable = 0 / En-able = 1	1 Bit	RWC---	[1.003] DPT_Enable

The scene function for this function block can be disabled by sending a 0 to this object					
	[A2] Timer 1 trigger	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is to trigger the first timer					
	Timer 1 change factor/Remaining time	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
Change factor: With this object the ON time of the timer can be changed. If the base is equal to 1 second, this object will change the time in seconds. If the base is 1 minute the value sent to the object is equal to the minutes the staircase will be ON, etc. Remaining time: Additionally to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value. In order to disable this function, the “T” flag must be deactivated. .					
	[A2] Timer 1 warning pulse	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
An additional object can be activated to send a warning pulse to inform that the staircase is about to expire and therefore have time to react in order to trigger it again.					
	[A2] Timer 1 disable	< Disable = 0 / Enable = 1	1 Bit	RWCT--	[1.003] DPT_Enable
With this object the timer will be disabled by receiving a 0					
	[A2] Timer 2 trigger	< On / Off	1 Bit	-WC---	[1.001] DPT_Switch
This is to trigger the second timer					
	[A2] Timer 1 change factor/Remaining time	< 1 byte unsigned	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
Change factor: With this object the ON time of the timer can be changed. If the base is equal to 1 second, this object will change the time in seconds. If the base is 1 minute the value sent to the object is equal to the minutes the staircase will be ON, etc. Remaining time: Additionally to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value. In order to disable this function, the “T” flag must be deactivated.					
	[A2] Timer 2 warning pulse	> On / Off	1 Bit	R-CT--	[1.1] DPT_Switch
An additional object can be activated to send a warning pulse to inform that the staircase is about to expire and therefore have time to react in order to trigger it again.					
	[A2] Timer 2 disable	< Disable = 0 / Enable = 1	1 Bit	RWCT--	[1.003] DPT_Enable
With this object the timer will be disabled by receiving a 0					
	[A2] Disable function block	< On / Off	1 Bit	RWCT--	[1.003] DPT_Enable
The function block can be disabled by this object. In the parameters one can decide to disable with a 1 or a 0.					
BINARY INPUT OBJECTS					
	[In1] Disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable

This is to disable the first input by sending a 1 to this object.					
	[In1] Disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
This is to disable the first input by sending a 0 to this object.					
	[In1] Switching short	> On / Off	1 Bit	RWCT--	[1.1] DPT_Switch
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching short	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching short	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching short	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching short	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching short	> 4 bytes float	4 Bytes	R-CT--	[14] 14.xxx
This is the action to be sent to the bus when pressing the button short. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> On / Off	1 Bit	RWCT--	[1.1] DPT_Switch
This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> 4 bytes float	4 Bytes	R-CT--	[14] 14.xxx
This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Switching long	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount

This is the action to be sent to the bus when pressing the button long. (The time for long operation can be configured in the parameters)					
	[In1] Multiple op. 1 pulse	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the first multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 1 pulse	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the first multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 1 pulse	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the first multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 1 pulse	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the first multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 2 pulses	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the second multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 2 pulses	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the second multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 2 pulses	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the second multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 2 pulses	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the second multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 3 pulses	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the third multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 3 pulses	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the third multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 3 pulses	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the third multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 3 pulses	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the third multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 4 pulses	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch

This is the fourth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 4 pulses	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the fourth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 4 pulses	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the fourth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 4 pulses	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the fourth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 5 pulses	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the fifth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 5 pulses	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the fifth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 5 pulses	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
This is the fifth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. 5 pulses	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
This is the fifth multiple operation object. The number of pulses to trigger this object can be changed in the parameters. Also the time between pulses and the value to be sent can be changed in the parameters.					
	[In1] Multiple op. long	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
It is also possible to configure for the multiple operation a time for long operation. If the button is pressed longer than this time this object will send the parametrized value					
	[In1] Multiple op. long	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
It is also possible to configure for the multiple operation a time for long operation. If the button is pressed longer than this time this object will send the parametrized value					
	[In1] Multiple op. long	> 0..100%	1 Byte	R-CT--	[5.1] DPT_Scaling
It is also possible to configure for the multiple operation a time for long operation. If the button is pressed longer than this time this object will send the parametrized value					
	[In1] Multiple op. long	> 2 bytes float	2 Bytes	R-CT--	[9] 9.xxx
It is also possible to configure for the multiple operation a time for long operation. If the button is pressed longer than this time this object will send the parametrized value					
	[In1] Flashing	> On / Off	1 Bit	R-CT--	[1.001] DPT_Switch
This is the object to send the flashing sequence to the bus. The ON and OFF time can individually be adjusted in the parameters.					
	[In1] Dimming on/off	> On / Off	1 Bit	-WCT--	[1.1] DPT_Switch

This is the ON/OFF telegram generated when pressing the button short if the input is configured to have a dimming function.					
	[In1] Dimming +/-	> 4 bits relative dimming	4 Bit	-WCT--	[3.7] DPT_Control_Dimming
This is the 4 bit relative dimming telegram generated when pressing the button long if the input is configured to have a dimming function. The step size and whether or not a stop telegram must be set can be configured in the parameters.					
	[In1] Blind move	> Up = 0 / Down = 1	1 Bit	-WCT--	[1.8] DPT_UpDown
This object is to move the blinds up or down according to the KNX DPT 1.008 with a long press of the button					
	[In1] Blind stop/step	> Step Up = 0 / Step Down = 1	1 Bit	-WCT--	[1.007] DPT_Step
This object is to move the slats up or down or to stop the blind according to the KNX DPT 1.007 with a short press of the button					
	[In1] Scene	> Sc1 (0=Play 128=Rec)... Sc64	1 Byte	--CT--	[18.001] DPT_Scene_control
This sends the scene number to the bus with a short press of the button and send a record telegram with a long press of the button.					
	[In1] Sequence output 1	> On / Off	1 Bit	-WCT--	[1.001] DPT_Switch
This is the first (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 1	> 1 byte unsigned	1 Byte	-WCT--	[5.10] DPT_Value_1_Ucount
This is the first (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 1	> 0..100%	1 Byte	-WCT--	[5.1] DPT_Scaling
This is the first (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 1	> 2 bytes float	2 Bytes	-WCT--	[9] 9.xxx
This is the first (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 2	> On / Off	1 Bit	-WCT--	[1.001] DPT_Switch
This is the second (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 3	> On / Off	1 Bit	-WCT--	[1.001] DPT_Switch
This is the third (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					
	[In1] Sequence output 4	> On / Off	1 Bit	-WCT--	[1.001] DPT_Switch
This is the fourth (out of max. 4) sequence output object of the first input and will send a value to the bus depending on the parametrized value. Depending on the type of sequence the output objects will sequentially switch ON or OFF (increment/decrement)					

	[In1] Sequence trigger	< On = Trigger / Off = Nothing	1 Bit	-WC---	[1.001] DPT_Switch
The sequence can be triggered from the bus with this object. This will do the same as if the input button is pressed.					
	[In1] Sequence trigger inverted	< On = Trigger inv. / Off = No	1 Bit	-WC---	[1.001] DPT_Switch
The sequence can be inverted from the bus with this trigger object.					
	[In1] Counter	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the output object to send the current counter value of this input to the bus. The counter can increase its value on rising and/or falling edge.					
	[In1] Counter	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This is the output object to send the current counter value of this input to the bus. The counter can increase its value on rising and/or falling edge.					
	[In1] Counter	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This is the output object to send the current counter value of this input to the bus. The counter can increase its value on rising and/or falling edge.					
	[In1] Counter threshold	< Reading/writing threshold	1 Byte	RWC---	[5.10] DPT_Value_1_Ucount
This object is to read/write the threshold value of the counter					
	[In1] Counter threshold	< Reading thresh-old	1 Byte	R-C---	[5.10] DPT_Value_1_Ucount
This object is to only read the threshold value of the counter					
	[In1] Counter threshold	< Reading/writing threshold	2 Bytes	RWC---	[7.1] DPT_Value_2_Ucount
This object is to read/write the threshold value of the counter					
	[In1] Counter threshold	< Reading thresh-old	2 Bytes	R-C---	[7.1] DPT_Value_2_Ucount
This object is to only read the threshold value of the counter					
	[In1] Counter threshold	< Reading/writing threshold	4 Bytes	RWC---	[12.1] DPT_Value_4_Ucount
This object is to read/write the threshold value of the counter					
	[In1] Counter threshold	< Reading thresh-old	4 Bytes	R-C---	[12.1] DPT_Value_4_Ucount
This object is to only read the threshold value of the counter					
	[In1] Counter alarm	> 1=Alarm, 0=No, < 0=Reset	1 Bit	RWCT--	[1.001] DPT_Switch
This sends an alarm message if the threshold of the counter has been reached.					
	[In1] Counter reset	< On = Reset / Off = Nothing	1 Bit	-WC---	[1] 1.xxx
With this object the counter can be reset. If the threshold has been reached the 1 bit “Counter alarm” object will be equal to “1” indicating alarm. This alarm object will reset to zero (when receiving a “1” on this “[In1] Counter reset” object, but it will not be sent to the bus.					

	[In1] Counter last value	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the last value of the counter at reset					
	[In1] Counter last value	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This is the last value of the counter at reset					
	[In1] Counter last value	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This is the last value of the counter at reset					
	[In1] Counter trigger input	< On = Trigger / Off = Trigger	1 Bit	-WC---	[1.001] DPT_Switch
The counter can also be triggered with a telegram from the bus. This will trigger the counter when receiving OFF and ON telegrams					
	[In1] Counter trigger input	< On = Nothing / Off = Trigger	1 Bit	-WC---	[1.001] DPT_Switch
The counter can also be triggered with a telegram from the bus. This will trigger the counter when receiving OFF telegrams					
	[In1] Counter trigger input	< On = Trigger / Off = Nothing	1 Bit	-WC---	[1.001] DPT_Switch
The counter can also be triggered with a telegram from the bus. This will trigger the counter when receiving ON telegrams					
	[In1] Counter additional count.	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This object counts the same input signal, but it can have different trigger parameters, than the main counter. E.g. This additional counter can be used to get daily values by resetting the additional counter every 24 hours for instance.					
	[In1] Counter additional count.	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This object counts the same input signal, but it can have different trigger parameters, then the main counter. E.g. This additional counter can be used to get daily values by resetting the additional counter every 24 hours for instance.					
	[In1] Counter additional count.	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount
This object counts the same input signal, but it can have different trigger parameters, then the main counter. E.g. This additional counter can be used to get daily values by resetting the additional counter every 24 hours for instance.					
	[In1] Counter additional count. reset	< 1 = Reset, 0 = Nothing	1 Bit	-WC---	[1.015] DPT_Reset
This is to reset the additional counter with a 1					
	[In1] Counter additional count. last value	> 1 byte unsigned	1 Byte	R-CT--	[5.10] DPT_Value_1_Ucount
This is the object to store the last value of the additional counter at reset.					
	[In1] Counter additional count. last value	> 2 bytes unsigned	2 Bytes	R-CT--	[7.1] DPT_Value_2_Ucount
This is the object to store the last value of the additional counter at reset.					
	[In1] Counter additional count. last value	> 4 bytes unsigned	4 Bytes	R-CT--	[12.1] DPT_Value_4_Ucount

This is the object to store the last value of the additional counter at reset.					
	[In1] MD lighting output	> On / Off	1 Bit	--CT--	[1.1] DPT_Switch
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting output	> 1 byte unsigned	1 Byte	--CT--	[5.10] DPT_Value_1_Ucount
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting output	> 0..100%	1 Byte	--CT--	[5.1] DPT_Scaling
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting output	> 2 bytes float	2 Bytes	--CT--	[9] 9.xxx
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting output	> 4 bytes float	4 Bytes	--CT--	[14] 14.xxx
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting output	> 4 bytes unsigned	4 Bytes	--CT--	[12.1] DPT_Value_4_Ucount
This object will send the parametrized lighting output value when the movement detector detects a movement.					
	[In1] MD lighting LUX input	< 2 bytes float	2 Bytes	RWC---	[9.4] DPT_Value_Lux
When configured to switch the light ON or OFF depending on the brightness by an additional object, this object is used to receive the brightness value from the bus.					
	[In1] MD lighting disable 1	< Disable = 1 / Enable = 0	1 Bit	-WC---	[1.003] DPT_Enable
This is the first lighting disable input object and will disable the movement detector when receiving a 1. This object only is an input object and does not reflect the status whether or not it is blocked, for that there is an additional status object.					
	[In1] MD lighting disable 1	< Disable = 0 / Enable = 1	1 Bit	-WC---	[1.003] DPT_Enable
This is the first lighting disable input object and will disable the movement detector when receiving a 0. This object only is an input object and does not reflect the status whether or not it is blocked, for that there is an additional status object.					
	[In1] MD lighting disable 2	< Disable = 0 / Enable = 1	1 Bit	-WC---	[1.003] DPT_Enable
This is the second lighting disable input object and will disable the movement detector when receiving a 1. This object only is an input object and does not reflect the status whether or not it is blocked, for that there is an additional status object.					
	[In1] MD lighting disable 2	< Disable = 1 / Enable = 0	1 Bit	-WC---	[1.003] DPT_Enable
This is the second lighting disable input object and will disable the movement detector when receiving a 0. This object only is an input object and does not reflect the status whether or not it is blocked, for that there is an additional status object.					
	[In1] MD lighting status	> Disable = 1 / Enable = 0	1 Bit	R-CT--	[1.003] DPT_Enable
This is the status telegram to indicate if the lighting channel of the detector is blocked or not. The value of the will be 1 when the channel is disable and a 0 when enabled					

	[In1] MD HVAC output	> On / Off	1 Bit	--CT--	[1.1] DPT_Switch
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC output	> 0..100%	1 Byte	--CT--	[5.1] DPT_Scaling
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC output	> 1 byte unsigned	1 Byte	--CT--	[5.10] DPT_Value_1_Ucount
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC output	> 2 bytes float	2 Bytes	--CT--	[9] 9.xxx
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC output	> 4 bytes float	4 Bytes	--CT--	[14] 14.xxx
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC output	> 4 bytes unsigned	4 Bytes	--CT--	[12.1] DPT_Value_4_Ucount
This is the HVAC output object for the movement detector and will send the parametrized value to the bus depending of the settings in the parameters. By default it will not immediately send a telegram on detection, but only after detecting for a set time.					
	[In1] MD HVAC disable	< Disable = 1 / Enable = 0	1 Bit	RWC---	[1.003] DPT_Enable
This will disable the HVAC channel when receiving a 1					
	[In1] MD HVAC disable	< Disable = 0 / Enable = 1	1 Bit	RWC---	[1.003] DPT_Enable
This will disable the HVAC channel when receiving a 0					
	Alarm 1 status	> ON = Alarm, OFF = No alarm	1 Bit	R-CT--	[1.005] DPT_Alarm
This is the alarm 1 status object and it will indicate with a 1 if there is an alarm and send a 0 if there is no alarm					

3 Parameter page

3.1 Parameter page: General Settings

Parameter	Settings
Device Name	**InBlock**
Here a personalized name for each device can be entered. E.g. **InBlock living room**	
Inputs	No **Yes**
Use this parameter to activate or deactivate all input parameters and their objects.	
ADVANCED FUNCTIONS	
All advanced features of the InBlock actuator can be activated or hidden as desired. It also serves as useful overview of all the functions available. These functions are totally inputs independent. You could even deactivate the inputs totally, thus converting the device into a pure controller module	
Function Blocks	**No** Yes
Use this parameter to activate or deactivate all function blocks parameters and their objects.	
Alarms	**No** Yes
Use this parameter to activate or deactivate all alarm parameters and their objects.	
Logics	**No** Yes
Use this parameter to activate or deactivate all logic parameters and their objects.	
Scene controller	**No** Yes
Use this parameter to activate or deactivate all scene controller parameters and their objects.	
Timers	**No** Yes
Use this parameter to activate or deactivate all timer parameters and their objects.	
Setpoints	**No** Yes
Use this parameter to activate or deactivate all setpoint parameters and their objects.	
Internal variables	**No** Yes
Use this parameter to activate or deactivate all parameters for the internal variables.	

Overwrite end-user parameter values at download	No **Yes** Custom
By selecting “no” the end-user parameters will not be overwritten when downloading the application with the ETS. When selecting Custom the “ENDUSER PARAMETERS” tab will be activated in which almost each end-user parameter can be individually selected whether to overwrite or not.	
Central sending object for monitoring device	**No** Yes
Use this parameter to activate or deactivate the “Central cyclic telegram for monitoring” object. This object will send a cyclic ON telegram to the bus in order to supervise the device.	
Behaviour at bus recovery	**No** Yes
Use this parameter to activate or deactivate the behaviour at bus recovery.	

4 Parameter page: InX Inputs

There are 6 inputs which can be configured to receive binary (push buttons, window contacts, water leakage sensor…) and analog signals (movement detector, temperature sensor and monitored input…)

Parameter	Settings
Input 1	No function **Binary input** Movement detector

Parameter page: InX Binary input

Parameter	Settings
Type of input	**Switching / value** Dimming Shutter KNX Scene Multiple operations Flashing Sequence Counter

4.1 Parameter page: Binary input / Switching / value

Parameter	Settings
Type of input	**Switching / value**
To send values to the bus depending of the next parameters.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	

4.1.1 Parameter page: Switching / value / operation mode

Parameter	Settings

Type of switching function	**Short operation** Short + Long operation Short + Long operation advanced
This parameter is to select the way the input will be operated. With Short operation one can have different events for rising and falling edge. Whereas with the other two selections the events for short and long operation can be selected.	

4.1.2 Parameter page: Switching / value / Short operation

Parameter	Settings
Type of switching function	**Short operation**
Here one can have different events for “Event on closing the contact” rising edge and “Event on opening the contact” falling edge.	
Datapoint type short operation object	**1 bit** 1 byte scaling 1 byte unsigned 2 bytes float 4 bytes unsigned 4 bytes float
Here the Datapoint type for the short operation object can be selected.	
Event on closing the contact	**Toggle** On Off No function
A telegram with one of the above options (if DPT=1 bit where Toggle = opposite to the objects value) as its useful data will be sent when closing the contact. (rising edge) By changing the DPT the value to be sent can be introduced in an input field and the possible range depends on the DPT selection. For 2 byte float values the introduced value will be multiplied by 0.1 in order to send decimal values.	
Event on opening the contact	Toggle On Off **No function**
A telegram with one of the above options (if DPT=1 bit where Toggle = opposite to the objects value) as its useful data will be sent when opening the contact. (falling edge) By changing the DPT the value to be sent can be introduced in an input field and the possible range depends on the DPT selection. For 2 byte float values the introduced value will be multiplied by 0.1 in order to send decimal values.	
Delay of telegram	**No** At closing At opening Both
The telegram can be delayed from 1 to 255s for any of the above options.	
Cyclic sending for	**No** Closing Opening Both

The telegram can be repeated cyclically for any of the above options. Whether or not the cyclic sending can be stopped with by enabling and/or disabling the input can also be configured.	
Send input status after bus recovery	**No** Yes
The last input status can be saved on bus voltage failure and will be sent to the bus (the initial sending delay can be adjusted in the general setting tab) on bus voltage recovery if yes is selected.	

4.1.3 Parameter page: Switching / value / Short + Long operation

Parameter	Settings
Type of switching function	Short + Long operation advanced
Attention! Advanced = event for short + event for long + event for opening after long	
SHORT OPERATION	No **Yes**
This parameter is to activate the short operation	
Datapoint type short operation object	**1 bit** 1 byte scaling 1 byte unsigned 2 bytes float 4 bytes unsigned 4 bytes float
Here the Datapoint type for the short operation object can be selected.	
Event on short operation	**Toggle** On Off
A telegram with one of the above options (if DPT=1 bit where Toggle = opposite to the objects value) as its useful data will be sent when opening the contact before the time for long operation has elapsed. By changing the DPT the value to be sent can be introduced in an input field and the possible range depends on the DPT selection. For 2 byte float values the introduced value will be multiplied by 0.1 in order to send decimal values.	
LONG OPERATION	No **Yes**
This parameter is to activate the long operation	
Datapoint type long operation object	**1 bit** 1 byte scaling 1 byte unsigned 2 bytes float 4 bytes unsigned 4 bytes float
Here the Datapoint type for the long operation object can be selected.	
Event on long operation	Toggle **On** Off
A telegram with one of the above options as its useful data will be sent when opening the contact after the time for long operation has elapsed.	
Time for long operation	**100 ms** 1 s

This time is to distinguish between short and long operation. When releasing before this time, the short operation event will be executed, and afterwards the event for the long operation will be sent.	
OPENING CONTACT	No **Yes**
(Only for “Switching / value / Short + Long operation advanced”) This parameter is to activate the event for opening the contact after the time for long operation has elapsed.	
Event on opening the contact after long operation	**Toggle** On Off
A telegram with one of the above options (if DPT=1 bit where Toggle = opposite to the objects value) as its useful data will be sent when opening the contact after the time for long operation has elapsed. By changing the DPT the value to be sent can be introduced in an input field and the possible range depends on the DPT selection. For 2 byte float values the introduced value will be multiplied by 0.1 in order to send decimal values. Attention! This event will be delayed by 50ms and sent using the same object as for long operation	
Delay of telegram	**No** At short operation At long operation At opening contact At all operations
The telegram can be delayed from 1 to 255s for any of the above options.	
Cyclic sending	**No** Short operation Opening contact after long operation Last operation
The telegram can be repeated cyclically for any of the above options. Whether or not the cyclic sending can be stopped with by enabling and/or disabling the input can also be configured.	

4.2 Parameter page: Binary input / Dimming / General Settings

Parameter	Settings
Type of input	Dimming
Select this option to dim a light connected to a KNX dimming actuator	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	

Attention! For 1 byte absolute dimming use the Sequence function	
Monitor input open circuit / Doubling inputs	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line. With the above options one can select what value (nothing, Off, On, Toggle) should be sent with an open circuit alarm and also what value (nothing, Off, On, Toggle) when the alarm goes away.	

Parameter	Settings
Function of input	Off / darker On / brighter **Toggle brighter / darker**
Select here the function of the input from one of the above options	

4.2.1 Parameter page: Dimming / Toggle brighter/darker

Parameter	Settings
Function of input	**Toggle brighter / darker**
With this selection the opposite event to the last executed/received event will be sent. e.g. Previous event: ON -> next event: OFF Previous event: Dim brighter -> next event: Dim darker And vice versa.	
Dimming direction after switching ON	**Darker** Brighter
After sending a ON with the 1 bit object, the next dimming event (4 bit dimming object) will send the parametrized dimming step with dimming direction equal to “Darker”	
Time for long operation	**100 ms** 1 s
This time is to distinguish between short and long operation. When releasing before this time, the 1 bit ON/OFF short operation event will be executed. When reaching this time the 4 bit dimming long operation event will be sent and afterwards when releasing either a stop telegram or not will be sent depending on the next parameter.	
Dimming step	**1 step (100%)** 2 steps (50%) 4 steps (25%) 8 steps (12,5%) 16 steps (6,25%) 32 steps (3,12%) 64 steps (1,6%)

A dimming command, relative to the current brightness setting, is transmitted to the dimming actuator using the relative dimming object DPT_Control_Dimming. Bit 3 of the useful data determines whether the addressed device dims down or up compared to the current brightness value. Bits 0 to 2 determine the dimming step. The smallest possible dimming step is 1/64th of 100 % (1 % in the ETS group monitor).	
Send stop telegram when opening contact	No **Yes**
By selecting this option a stop telegram will be sent when releasing after passing the “time for long operation”	
Cyclic sending	**No** Yes
The telegram will be repeated cyclically (with a configurable frequency), but only during the time the contact is closed.	

4.2.2 Parameter page: Dimming / Off / darker

4.2.3 Parameter page: Dimming / On / brighter

Parameter	Settings
Function of input	Off/ darker On / brighter
Select the function of the input to switch ON with a short operation and dim brighter with a long operation or switch OFF with a short operation and dim darker with a long operation	
Time for long operation	**100 ms** 1 s
This time is to distinguish between short and long operation. When releasing before this time, the 1 bit ON/OFF short operation event will be executed, and afterwards the 4 bit dimming long operation event will be sent.	
Dimming step	**1 step (100%)** 2 steps (50%) 4 steps (25%) 8 steps (12,5%) 16 steps (6,25%) 32 steps (3,12%) 64 steps (1,6%)
A dimming command, relative to the current brightness setting, is transmitted to the dimming actuator using the relative dimming object DPT_Control_Dimming. Bit 3 of the useful data determines whether the addressed device dims down or up compared to the current brightness value. Bits 0 to 2 determine the dimming step. The smallest possible dimming step is 1/64th of 100 % (1 % in the ETS group monitor).	
Send stop telegram when opening contact	No **Yes**
By selecting this option a stop telegram will be sent when releasing after passing the “time for long operation”	
Cyclic sending	**No** Yes

The telegram will be repeated cyclically (with a configurable frequency), but only during the time the contact is closed.

4.3 Parameter page: Binary input / Shutter

Parameter	Settings
Type of input	Shutter
Select this option to control a shutter connected to a KNX shutter actuator	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function, the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.3.1 Parameter page: Shutter / Blind

Parameter	Settings
Event on short operation	**Stop / step up** Stop / step down Toggle stop / step Up Down Toggle up / down
Here the event for the short operation can be assigned. Take note that any of the events can be configured, unlike most KNX shutter/blind sensors.	
Event on long operation	Stop / step up Stop / step down Toggle stop / step **Up** Down Toggle up / down

Here the event for the long operation can be assigned. Take note that any of the events can be configured, unlike most KNX shutter/blind sensors.	
Time for long operation	**100 ms** 1 s
This time is to distinguish between short and long operation. When releasing before this time, the short operation event will be executed, and afterwards the event for the long operation will be sent. Take note that any of the events can be configured for both short and long operation and therefore the objects only indicate the event and not if it is for short or long. I.e. If event for short operation = UP and event for long operation = Down, the “[InX] Blind stop/step” object will never send a telegram.	
Slat time push button	**No** Yes
This is to send a stop telegram after long operation and when releasing within the parametrized time. After this time no telegram will be sent This time should be longer than the total slat time configured in the shutter/blind output channels.	
Waiting time to change slat direction (between short step actions) * Only for Toggle	**100 ms** 1 s
This time is essential to move the slats (with repeated short events) in the same direction when “Toggle” is selected. With short step actions longer than this time the next short event will be the inverted action.	
Attention! This time must be longer than the time configured for long operation * Only for “Event on short operation” = Toggle up / down	

4.4 Parameter page: Binary input / KNX Scene / General Settings

Parameter	Settings
Type of input	KNX Scene
This type of input selection assigns the input to be a standard KNX 8 bit DPT_Scene_Control sensor.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with a ON telegram and to disable with an OFF telegram or vice versa.	
Execute scene after bus recovery	**No** Yes
With this option the scene will be executed (the initial sending delay can be adjusted in the general setting tab) on bus voltage recovery.	

Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.4.1 Parameter page: KNX Scene

Parameter	Settings
Scene number	**Scene 1** … Scene 64
The scene number to be sent can be configured here. Scene 1 = value 0, Scene 2 = value 1 and so forth up to value Scene 64 = value 63.	
Save scene with long operation	No **Yes**
With this selection the scene can be saved. Saving Scene 1 will send the value 128, Scene 2 sends value 129 and so forth up to Scene 64 sends value 191 to the bus.	
Time for long operation	**100 ms** 1 s
This time is to distinguish between short and long operation. When releasing before this time, the scene will be executed, and afterwards the scene will be saved.	

4.5 Parameter page: Binary input / Multiple operations

Parameter	Settings
Type of input	Multiple operations
With this option more than one telegram can be sent with the same input depending on the number of pulses.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with a ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms

This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function, the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.5.1 Parameter page: Multiple operations / Operation 1…5

Parameter	Settings
Multiple operation 1 (1…5)	No **Yes**
A total of 5 multiple operation can be activated one by one by selecting yes in each one.	
Number of pulses	**1 pulse** … 10 pulses
The number of pulses in the input to execute an event as configured in the next parameters	
Datapoint type of output	**1 bit** 1 byte unsigned 1 byte scaling 2 bytes float
Here the Datapoint type for the “[InX] Multiple op. X pulses”] object can be selected.	
Action on X pulses	On Off **Toggle**
A telegram with one of the above options as its useful data will be sent as the Action on the above configured number pulses.	
Maximum time between pulses	**500 ms** 1 s 2 s 5 s 10 s
For the pulses to be counted, the time between the consecutive pulses may not exceed this parametrized maximum time. Should the time between two consecutive pulses exceed this time, this last pulse and all the following pulses will not be taken into account. It will only start to execute the pulses again once all other multiple operations for this input has been executed.	
Condition for sending value	**Only evaluate last executed pulse operation** Evaluate immediately when operations = pulses

Configure here the sending condition of the output. When “Only evaluate last executed pulse operation” has been selected, the output object will only be sent when the last pulse (when the maximum time between pulses has elapsed) is equal to the number of configured pulses. When “Evaluate immediately when operations = pulses” has been selected, when the number of operations equals the number of pulses, the output will be immediately sent. It will not wait for the last pulse (when the maximum time between pulses has elapsed) to be executed.

4.5.2 Parameter page: Multiple operations / Long operation

Parameter	Settings
Long operation	No **Yes**
This activates the long operation	
Time for long operation	**100 ms** 1 s
This time is to distinguish between pulses and long operation. When releasing before this time, a pulse is counted, and afterwards event for long will be executed.	
Datapoint type for long operation output	**1 bit** 1 byte unsigned 1 byte scaling 2 bytes float
Here the Datapoint type for the “[InX] Multiple op. long object” can be selected.	
Event on long operation	**Toggle** On Off
A telegram with one of the above options as its useful data will be sent when opening the contact after the time for long operation has elapsed.	

4.6 Parameter page: Binary input / Flashing / General Settings

Parameter	Settings
Type of input	Flashing
The input can be used to flash ON and OFF with different ON and OFF times.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with a ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms

This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function, the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.6.1 Parameter page: Flashing

Parameter	Settings
Flashing	Close = flash, open = nothing Close = nothing, open = flash **Close = flash, open = stop** Close = stop, open = flash Both = start flashing
Select here with which operation (by opening the contact or closing the contact) the flashing should start and stop. Take into account that the flashing will only start if the contact is opened or closed while the device has bus voltage. Should the contact be closed while there is no bus voltage, and the bus voltage recovers afterwards, then the flashing will neither start nor stop.	
ON duration	1 s 5 s 10 s 1 m 5 m 10 m 1 h
The ON duration can be configured here	
OFF duration	1 s 5 s 10 s 1 m 5 m 10 m 1 h
The OFF duration can be configured here	
Number of repetitions (65535 = always flashing)	65533
This is the number of repetitions the ON/OFF flashing sequence should perform. 0 = No repetitions and 65535 = always flashing.	
Stop flashing	**No** At disabling input At disabling and enabling input
The flashing can be stopped either only at disabling or both for enabling and disabling the input.	

4.7 Parameter page: Binary input / Sequence / General Settings

Parameter	Settings
Type of input	Sequence
With this option loads can be sequentially switched ON or OFF. This can be used to have for instance more or less lights ON and thus create the illusion of “dimming” the lights with normal switching actuators.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function, the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.7.1 Parameter page: Sequence

The sequence is to switch from one to four output objects sequentially ON or OFF. The sequence is triggered with the rising edge of the input.

Parameter	Settings
Datapoint type of sequence objects	**1 bit** 1 byte unsigned 1 byte scaling 2 bytes float
The datapoint type of the sequence objects can be selected here.	
Number of sequence objects	4
The number of the sequence object can be selected here.	
Type of sequence	Single **Multiple**
The type of the sequence can be selected here. When selecting “Single” only one sequence output object is ON at a time and when selecting “Multiple” more than one object can be ON at a time.	
Multiple (switch sequentially output objects ON)	Incremental ON loop Incremental ON

	Decremental OFF Decremental OFF loop **Toggle pause** Toggle

Select here in which order the output objects should be switched.

Incremental ON loop:
1>1+2>1+2+3>1+2+3+4>All OFF>1>1+2>1+2+3>…
Incremental ON loop:
1>1+2>1+2+3>1+2+3+4>stay in 1+2+3+4
Decremental OFF:
4+3+2+1>3+2+1>2+1>1>OFF>stay in OFF
Decremental OFF loop:
4+3+2+1>3+2+1>2+1>1>OFF>4+3+2+1>3+2+1>…
Toggle pause:
(1>1+2>1+2+3>1+2+3+4>Off>1…) pause > 1,5sec. (4+3+2+1>OFF>4>…)
The pause time for “Toggle pause” is equal to 1.5 sec. which means that with short pulses less than 1.5 sec. apart it will sequentially switch ON and after waiting more than this time it will sequentially switch OFF.

Toggle:
Off>1>1+2>1+2+3>1+2+3+4>1+2+3>1+2>1>Off>1+2>...

Single (only one object ON at a time)	Incremental loop Incremental **Toggle pause** Toggle Decremental Decremental loop

Toggle pause (1>2>3>4>Off>1…), (4>3>2>1>OFF>4>…)
Attention! Pause time for “Toggle pause” = 1,5 sec.

Incremental loop:
1>2>3>4>Off>1>…
Incremental:
Off>1>2>3>4>stay in 4
Toggle pause:
(1>2>3>4>Off>1>…) pause > 1,5sec. (4>3>2>1>Off>4>…)
The pause time for “Toggle pause” is equal to 1.5 sec. which means that with short pulses less than 1.5 sec. apart it will sequentially switch ON (only one at a time) and after waiting more than this time it will sequentially switch OFF.
Toggle:
Off>1>2>3>4>3>2>1>Off>1>…
Decremental
4>3>2>1>stay in Off
Decremental loop
4>3>2>1>Off>4>…

Objects to send	**All objects** Only changed objects

It can be selected whether only changed objects or all objects should be sent on each operation.

Additional input object to trigger sequence (only ON)	**No** Yes

The sequence can also be triggered from the bus to do the same as if the input was pressed. It will only be triggered with ON telegrams.	
Additional input object to inverse sequence (increment / decrement)	**No** Yes
This activates an object to inverse the selected sequence. If the input is used to increment the sequence, with this object the same sequence can be decremented form the bus. It will only be triggered with ON telegrams.	

4.8 Parameter page: Binary input / Counter

Parameter	Settings
Type of input	Counter
With this parameter the input can be used as a counter.	
Enable / Disable input	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The input can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Send counter values after bus recovery	**No** Yes
The last counter value can be saved on bus voltage failure and will be sent to the bus (the initial sending delay can be adjusted in the general setting tab) on bus voltage recovery if yes is selected.	
Debounce time	10 ms 20 ms **50 ms** 100 ms 150 ms 200 ms
This parameter is used to set the time the input will be blocked after receiving an input signal. This ensures that the input does not generate unwanted duplicate telegrams.	
Monitor input open circuit / Doubling input	**No** Alarm = 1, No alarm = 0 Alarm = 0, No alarm = 1 Alarm = Toggle, No alarm = X No alarm = Toggle, Alarm = X
By selecting this function, the inputs can be supervised in order to generate an alarm if the input connexion has been cut (only open circuit will generate an alarm). To do this a 2,7k Ohm resistor must be connected to the end of the input line.	

4.8.1 Parameter page: Counter / No / Upward / Backward

Parameter	Settings
Counter	**No** Upward Backward
There two types of counters; Upward = counts up on each trigger event and Backward = counts backward on each trigger event	

Parameter	Settings

Counter	Upward
Counts up on each trigger event	
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**
Here the datapoint type for the counter can be selected. Usually, a Switching counter has a 4 bytes unsigned (default option) value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values. Attention: Should the counter be programmed with one DPT and in a later stage the DPT is changed the conter value will be overwritten to zero or to the “Initial value counter”	
Count number of triggers on	**Rising edge** Falling edge Rising and falling edge
Decide here the trigger events to increase or decrease the counter. With rising edge, the counter will only be triggered when closing the input. With falling edge, the counter will only be triggered when opening the input. And with rising and falling edge the counter will be triggered both when closing and opening the input.	
Additional inputs object to trigger counter	**No** Only with ON Only with OFF Both
The counter can also be triggered from the bus. Depending on this parameter the counter will be triggered with ON telegrams, OFF telegrams, or with both.	
Initial value counter	**No** Yes
Here the initial different starting value of the counter can be configured. After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter will also be reset. Practical example: should the actuator be installed in an existing installation, where the load connected to the current channel has already a known number of switching operations, this information can be used as the “New starting value”. But in a later stage, if some other parameter in the actuator must be changed and downloaded, the new current counter value will not be overwritten.	
Threshold value ***Attention! 0 = Deactivated***	**0**
Here you can enter the number of switching operations that will trigger the 1 bit alarm object of the current channel. So, this alarm object will be activated and send a “1” to the bus as soon as the switching counter passes this threshold. Attention, this alarm will also be sent to the bus immediately after bus recovery. Should the conversion factor be activated and set to be for example “Several triggers increases 1 step” = 3, and the threshold value is set to 5 then the sequence will be as follows: : 0,0,1,1,1,2,2,2,3,3,3,4,4,4,5,… The alarm is sent in the first 5 after 15 pulses.	
Object for reading / writing the threshold value	**No** Only readable Readable and writeable
With this option the threshold value can be read and/or changed from the bus.	

Only readable: this option will activate an unsigned counter object, which can be read by the ETS/other KNX devices. Readable and writable: this option will activate an unsigned counter object, which can be read and overwritten by the ETS/other KNX devices. This is meant to allow changing the threshold value with, for instance, a visualization. Should the threshold value be changed by the	
Reaction on overflow (Max. value of DPT)	**Reset to 0 and start again** Stay at maximum
Attention! Both counter & alarm objects will be set to zero	
Important note: the overflow must not be mistaken with the threshold value, since they are two totally different concepts: - An overflow is reached when the object value exceeds the maximum value of the selected data point type. For example, the maximum value of a 1 byte unsigned value is 255; therefore, the overflow is reached when the object value exceeds 255. - On the other hand, the threshold refers to any given value of your choice that is valid for this DPT. Reset to 0 and start again (default option): when then overflow is reached, the object will start counting from 0 again. Attention! In this case the alarm object will also be set to zero, otherwise one would not know if the threshold has newly been reached or not. Stay at maximum: in the event of the overflow being reached, the object will stop at the maximum value of the DPT.	
Additional functions	**No** Yes
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.	

Parameter	Settings
Counter	Backward
Counts backward on each trigger event	
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**
Here the datapoint type for the counter can be selected. Usually, a Run hour counter has a 4 bytes unsigned (default option) value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values	
Count number of triggers on	**Rising edge** Falling edge Rising and falling edge
Here can be decided when the counter should be triggered. When closing the contact (Rising edge), opening the contact (Falling edge) or both (Rising and falling edge)	
Additional inputs object to trigger counter	**No** Only with ON Only with OFF Both
The counter can also be triggered from the bus with the above options.	

Initial value counter	**800**
Attention! After programming this value will only be overwritten if the new starting value is changed	
Here the initial different starting value of the counter can be configured from which the counter will count back. It will send a 1 bit alarm telegram with the value “1” when reaching the value zero. Attention! This value will never be sent. The 1st value sent will be the first decreased value. After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter will also be overwritten if the main counter is overwritten. Should the conversion factor be activated and set to be for example “Several triggers decreases 1 step” = 3, and the “Initial value switching counter” is set to 5 then the sequence will be as follows: 444,333,222,111,000, and only at the last 0 the alarm will be sent.	
Reaction on reaching zero	Stay at zero **Reset to initial value and start again**
Stay at zero: once the counter reaches 0, it will stay there until it has been reset. Reset to initial value and start again (default option): once the counter reaches 0, it will start counting back again starting from the initial value of the switching counter (as parameterized in the previous option).	
Additional functions	**No** Yes
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.	

4.8.2 Parameter page: Counter / Additional functions

Parameter	Settings
Cyclic sending of counter value	**No** Yes
With this option the counter values can be sent cyclicly which can have a frequency from 10 sec. up to 255 hours.	
Counter values are sent to the bus every: (Triggers)	**1**
Enter here the number of switching operations that be executed before the counter sends its value to the bus. This option is meant to reduce the bus traffic. For instance, if you enter a “50”, the counter will send its first value whenever the accumulated switching operations of the channel amount to 50 and will then send the value 50 to the bus (50, 100, 150, 200, 250…).	
Conversion factor	**None** Several triggers increase 1 step 1 trigger increases several steps
None (default option): for each switching operation of the channel, the counter increases 1 step. Several triggers increase 1 step: define here the number of triggers that must be received for the counter to increase 1 step. Should it be set to the value 10, then only when triggers received amount to 10, will the counter increase 1 step. 1 trigger increases several steps: define here the step increment for each trigger received. For example, if it is set to 50, after 50 triggers received, the counter will have increased 50 x 10 (= 500) steps.	
Send last value of counter at reset by counter object	**No** Yes

No (default option): if you reset the counter by using the 1 bit reset object, the last value of the counter will not be sent to the bus by the counter object. Instead, a “0” will be sent to indicate it has been reset. Yes: if you reset the counter by using the 1 bit reset object, the counter object will send its current value before reset to the bus and afterwards it will not reset to 0 but stay at its last value. Only at the next counter step, will the first counter step be sent to the bus. Thus the counter will never have the value “0”.	
Additional object to store last value of counter on reset	**No** Yes Yes and send
No (default option): no additional object to store the last value of the counter on reset will be activated. Yes: an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). Yes and send: an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.	
Activate additional counter * Only with counter Upward	**No** Yes
The additional counter counts the same input signal. It can be used to inform about, for example, the daily value. To do this a time switch is needed to reset this additional counter once a day (or any other desired interval)	
Additional upwards counter	Rising edge **Falling edge** Rising and falling edge
Here can be decided when the additional counter should be triggered. When closing the contact (Rising edge), opening the contact (Falling edge) or both (Rising and falling edge)	
Additional upwards counter initial value	**0**
Here the initial different starting value of the counter can be configured from which the counter will count. After downloading with the ETS this value will only be overwritten if the new starting value is changed.	
Reaction on overflow (Max. value of DPT)	**Reset to 0 and start again** Stay at maximum
Important note: the overflow must not be mistaken with the threshold value, since they are two totally different concepts: An overflow is reached when the object value exceeds the maximum value of the selected data point type. For example, the maximum value of a 1 byte unsigned value is 255; therefore, the overflow is reached when the object value exceeds 255. On the other hand, the threshold refers to any given value of your choice that is valid for this DPT. **Reset to 0 and start again:** when then overflow is reached, the object will start counting from 0 again. Attention! In this case the alarm object will also be set to zero, otherwise one would not know if the threshold has newly been reached or not. **Stay at maximum:** in the event of the overflow being reached, the object will stop at the maximum value of the DPT.	
Additional object to store last value of counter on reset	**No** Yes Yes and send

No: no additional object to store the last value of the counter on reset will be activated.
Yes: an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse).
Yes and send: an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.

4.9 Parameter page: Binary input / Movement detector

There are 6 inputs which can be configured to receive binary (push buttons, window contacts, water leakage sensor…) and movement detector.

Parameter	Settings
Input 1...6	No function Binary input **Movement detector**

4.9.1 Parameter page: Movement detector/ General Settings

The input of the actuator can be used to connect any conventional binary movement detector with a N.O. relay output. It has up to two channels: one lighting channel and a HVAC channel.

Parameter	Settings
Type of movement detector	**Time in parameter** Time in detector
Attention! For binary detector, manually adjust the pulse time in external detector as short as possible!	
The type of detector basically determines whether or not the time should be adjusted in the detector or in the application program. When selecting “Time in detector”, there is no detection time parameter in the ETS application program and the time must be set in the detector (usually with a small time adjustment screw). When selecting “Time in parameter”, the time can be adjusted in the application program. For the binary detector the pulse time should be adjusted manually with the small time adjustment screw on the detector to be as short as possible since the time starts counting the moment the relay opens.	

4.9.1.1 Parameter page: Movement detector / Time in parameter

When selecting “Analog & Bin detector. Time in parameter”, the time can be adjusted in the application program. For the binary detector the pulse time should be adjusted manually with the small time adjustment screw on the detector to be as short as possible.

With this selection both the lighting and HVAC channels will be available. (With “Time in parameter” only the lighting channel can be used.)

Both the lighting channel and the HVAC channel can be activated.

Parameter	Settings
Lighting channel	No **Yes**
This parameter is used to activate the lighting channel tab and all its parameters.	
HVAC channel	**No** Yes
This parameter is used to activate the HVAC channel tab and all its parameters.	

Blocking time after end of detection	500 ms
Factor (1…255)	4
The detector can be blocked for a configurable time after end of detection; this time can be set here. This could be important depending on the load to be switched by the detector. Passive IR movement detectors detect moving heat, the detector detects any heat source which crosses the IR sectors of the detector. Since a light bulb is hot when switched on and cools down when switched off, it also generates moving heat and thus the detector can falsely interpret this to be a movement, after which the light would switch on again. This time is meant to avoid this conflict and should be adjusted depending on the heat generated by the bulb to be controlled and the distance to the detector.	

4.9.1.2 Parameter page: Movement detector / Time in parameter / Lighting tab

Parameter	Settings
Datapoint type lighting channel output	**1 bit** 1 byte scaling 1 byte unsigned 2 bytes float 4 bytes unsigned 4 bytes float
The DPT of the output object for the lighting channel can be set to any of the above DPTs.	
Event at beginning of detection	Nothing **Value**
Value to send	1
Here the value to be sent to the bus at the beginning of detection can be set. The option to send nothing is also available.	
Event at end of detection	Nothing **Value**
Value to send	0
Here the value to be sent to the bus at the end of detection can be set. The option to send nothing is also available.	
Total time after last detection (Time starts when relay opens)	**1 s** 10 s 1 min 10 min 1 h
Factor (1…255)	60
This is the time which must elapse without having received a detection pulse in the input from the connected detector, for it to trigger the event on end of detection.	
Cyclic sending	**No** Only on detection Only at the end of detection Both
Here one can choose the cyclic sending of the output telegram to be only on detection, only at end of detection or in both cases.	

Brightness dependent switching	**No** External object
The detector can switch the light dependent on the brightness value. This value can be received from a KNX light sensor by sending its value to the external object of the input.	
Threshold (detection is enabled when brightness is lower than)	**80**
Attention! Internal fixed hysteresis = 10%. (Ex. Threshold = 80; Unblock < 80 Lux; blocks > = 88 Lux)	
This option is only available when “External object” have been selected. When selecting “External object” the value can be sent from a KNX light sensor to the external object of the input. It can then block the detector if the brightness is higher than the parametrized threshold value set here. In this case, this lux threshold has an internal fixed hysteresis of 10 %, meaning that the detector will be blocked at the parameter value + 10% and unblocked at the parameter value. For example, during the day (high LUX level) the detector is blocked, as it gets dark enough to detect, (i.e. lower than the parameter value) it should enable the detector and stay enabled until the light level increases with 10% of this value.	
Enable / disable lightning channel	**No** Yes
It is possible to block the lighting channel with one or even two “Enable / disable …” objects. These objects are purely trigger objects to enable or disable the detector and it is NOT necessary to enable or disable both objects in order to enable or disable the detector. The last action received on these objects will determine the state of the detector. Therefore, they will not inform about whether or not the detector is blocked. For this purpose, there is an additional status object to inform about whether the detector is enabled or not. *Practical example*: a very typical requirement in a KNX installation is to be able to block the light in an ON state (for instance, during a meeting) but it is as important to block the light in an OFF state. (For instance, projector mode). That is why there are two objects to block the detector, each with a different behaviour when blocking and unblocking.	
Reaction on bus voltage recovery	**Enable** Disable Last object status
Here we can configure whether the lighting channel of the detector should be enabled or not on bus voltage recovery. It can also return to the status before bus failure.	
Enable lighting channel by object 1	**En = 1 / Dis = 0** En = 0 / Dis = 1
Attention! The “MD lighting Disable 1&2” objects don’t indicate the “disabled” status. The last object updated sets the state (independent of the other object)	
Here you can configure the value to enable or disable the detector with the first enable object.	
Send telegram when enabling lighting channel	**Don’t send** Value
Value to send	**1**
Use this parameter to set the value to be sent to the bus when enabling the channel with the first enable object. This telegram will be sent on each enable telegram (no need to change from the disabled state)	
Send telegram when disabling lighting channel	**Don’t send** Value
Value to send	**0**

Set here the value to be sent to the bus when disabling the channel with the first enable object. This telegram will be sent on each disable telegram (no need to change from the enabled state)	
Enable lighting channel by object 2	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
Attention! The “MD lighting Disable 1&2” objects don’t indicate the “disabled” status. The last object updated sets the state (independent of the other object)	
Configure with this parameter the value to enable or disable the detector with the second enable object.	
Send telegram when enabling lighting channel	**Don’t send** Value
Value to send	**1**
Use this parameter to set the value to be sent to the bus when enabling the channel with the second enable object. This telegram will be sent on each enable telegram (no need to change from the disabled state)	
Send telegram when disabling lighting channel	**Don’t send** Value
Value to send	**0**
Set here the value to be sent to the bus when disabling the channel with the second enable object. This telegram will be sent on each disable telegram (no need to change from the enabled state)	

4.9.1.3 Parameter page: Movement detector / Time in parameter / HVAC tab

Parameter	Settings
Datapoint type HVAC channel output	**1 bit** 1 byte scaling 1 byte unsigned 2 bytes float 4 bytes unsigned 4 bytes float
The DPT of the HVAC output object can be selected here.	
Initial waiting time for HVAC activation (time starts when relay closes)	1 s 10 s **1 min** 10 min 1 h
Factor	3

This is the initial waiting time which must elapse for the HVAC channel of the detector to detect movement.

This time starts to count when the relay of the external detector closes. Should a person only go into the detection range of the detector and immediately thereafter go out again, the HVAC channel of the detector will not detect movement.

Thus the HVAC system will only be switched to the desired operating mode if someone goes into the room and stays in this room longer than the configured time.

Due to the fact that this is usually a long time (3 minutes default parameter) and passive IR detectors are not perfect (they don't detect always all small movements, they only detect moving heat objects), a special algorithm has been implemented to determine if someone is staying in the room or not.

Explanation of this algorithm by means of an example: Let's say the "Initial waiting time…" is set to be 10 min. Then the first 50% (5min.) of the time, the detection pulses are ignored. Thereafter, during the rest of the time the input should detect detection pulses within a time window equal to 30% of the full "Initial waiting time…" (every 30% of 10min. = 3min.), otherwise the time will reset to the initial 10 minutes and the process will start all over again.

In other words, in this example:
During the first 5 minutes it will not detect any pulses.
From minute 5 to minute 8: the input must detect at least one pulse. If the pulse is received, it will reset the 30% timer.
The input detects a pulse at minute 6, then the input must detect the next pulse from minute 6 to minute 9.
The input detects a pulse at minute 7, then the input must detect the next pulse from minute 7 to minute 10.
Then the input detects a pulse just after minute 7, then the HVAC channel will be activated on minute 10 even if no pulse is afterwards received.

Event at beginning of detection	Nothing **Value**
Value to send	1
Configure here the value to be sent to the bus at the beginning of detection of the HVAC channel. The option to send nothing is also available.	
Event at end of detection	Nothing **Value**
Value to send	0
Configure here the value to be sent to the bus at the end of detection of the HVAC channel. The option to send nothing is also available.	
Total time after last detection (Time starts when relay opens)	1 s 10 s **1 min** 10 min 1 h
Factor (1…255)	30
This is the time which must elapse without any detection for the input to send the event at end of detection. This time starts to count at the beginning of detection and thus when the initial waiting time ends.	
Cyclic sending	**No** Only on detection Only at the end of detection Both

Here one can choose the cyclic sending of the output telegram to be only on detection, only at end of detection or in both cases.	
Enable / disable HVAC channel by object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The HVAC channel can be enabled or disabled with a 1 bit object. Here can be decided to enable with a 1 and disable with a 0 or vice versa.	
Reaction on bus voltage recovery	**Enable** Disable Last object status
Whether the HVAC channel of the detector will be active or not on bus voltage recovery can be configured here. On bus voltage recovery the HVAC channel can be enabled, disabled, or have the same state as before the bus failure depending on the above selection. **Enable:** the HVAC channel will be enabled. **Disable:** the HVAC channel will be disabled. **Last object status:** the status of the Enable object will be saved in the actuator's non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will set the object as it was before the bus failure.	
Send telegram when enabling HVAC channel	**Don't send** Value
Value to send	**0**
Use this parameter to define the value to be sent to the bus when enabling the HVAC channel with the HVAC enable object.	
Send telegram when disabling lighting channel	**Don't send** Value
Value to send	**0**
Use this parameter to define the value to be sent to the bus when disabling the HVAC channel with the HVAC enable object.	

4.9.1.4 Parameter page: Movement detector / Time in detector

When selecting "Time in detector" there is no detection time parameter in the ETS application program and the time must be set in the detector (usually with a small time adjustment screw). For this reason, only the lighting channel can be used.

All the parameters of the lighting channel are the same as in the previous type of movement detector, but without the parameter to adjust the time after last detection. There is no HVAC channel.

5 Parameter page: ADVANCED FUNCTIONS

Tip! REDUCE CONFIG TIME! All repetitive Tab & Sub-Tab parameters (Ex. “Function Block A1…X” or “Logic 1…X”…) can be changed at the same time by selecting multiple tabs with “CTRL + Click”.

Parameter	Settings
Function blocks	No **Yes**
The function blocks of the device are by default activated. Nevertheless, this device can also be used as an advanced controller module for logic functions, timers, etc. In this case, you can deactivate the function blocks totally and completely hide all their options and objects by selecting “No”.	

Parameter page: FUNCTION BLOCKS

Parameter	Settings
Function block A1 & A2 … Function Block D1 & D2	**No** Yes
Central ON/OFF object	**No** Yes
In order to do a classic KNX “Central function”, this actuator has a specific option that allows for all the function blocks inputs to receive at once with only one object. This considerably reduces the amount of group address associations (both meant to ease programmers work load, but also to reduce the actuator’s association table). Before we configure the function within the function block, we must activate the object.	

5.1 Parameter page: FUNCTION BLOCKS / A1…X1

Parameter	Settings
Invert input	**No** Yes
Use this parameter option to set whether the output relay closes with ON (“1”) and opens with OFF (“0”) or if it closes with OFF (“0”) and opens with ON (“1”).	
Input value on bus voltage failure	**Unchanged** ON OFF
Here you can select one of the following reactions**:** if “Unchanged”, whenever the bus voltage fails, the input value keeps the actual value. If you choose ON/OFF, ***as soon as the bus voltage fails, the input value is updated with an on/off***	
Input value on bus voltage recovery	**Unchanged** ON OFF Recovery status before bus failure Timer 1 reaction at ON Timer 2 reaction at OFF
Here you can select one of the following reactions**:** If “Unchanged”, whenever the bus voltage returns, the input value keeps the actual one. With ON/OFF, ***as soon as the bus voltage fails, the input value is updated with an on/off.*** With “Recovery status before bus failure**”,** the status of the output will be saved in the actuator’s non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will send the value to the function block output as it was before the bus failure. Each function block output has two timer functions. Only the first timer can be assigned to the reaction on bus voltage recovery. Timer 1 reaction at ON: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT ON” will be executed. Timer 1 reaction at OFF: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT OFF” will be executed.	
Advanced functions	No **Yes**
The InBlock device is also a powerful controller module (logic, timer, counter, etc. module). You can find Advanced Functions: In the General Settings parameter page: this a totally independent controller module, with its own input and output objects, which can work autonomously (no need to be linked to any actuator function). On top of that, the most common BINARY FUNCTIONS in Power Block series, are now included in the advanced functions named as FUNCTION BLOCKS.	

5.1.1 Paremeter page: FUNCTION BLOCK / A1…X1 / Output

Each function block has a separate tab to configure its output parameters, such as the different sending conditions.

Parameter	Settings
Send Output telegram	**Only on change** Always Only on change - Inverted Always - Inverted No
Only on change: the output of the function block will only be sent whenever the contact switches from on to off or vice versa. **Always:** after reception of each input function block telegram, the output will be sent to the bus. **Only on change – Inverted:** the inverted output will only be sent whenever the input changes from on to off or vice versa. **Always – Inverted:** after reception of each input value, the inverted output will be sent to the bus. **Only readable:** the “Output object” of this function block will be ready for sending its value after a read request.	
Cyclic sending Output telegram	**No** Only ON Only OFF Both ON / OFF
No: the Output telegram is only sent once. **Only ON:** if the Output changes to ON status, it will send the ON value cyclically. **Only OFF:** if the output changes to OFF status, it will send the OFF value cyclically. **Both ON / OFF:** in both cases (when the output changes to ON or OFF value), it will send the corresponding value cyclically. For these last three options the cyclic sending time can have a base of 10s, 1 min, 5 min, 10 min, 1 hour, and the factor can be from 1 to 255. Should an output telegram be sent (not because of cyclic sending) the cyclic sending time will be reset in order to avoid unwanted duplicate telegrams.	
Delay Output telegram	**No** Yes
Depending on the previously configured sending condition, the Output telegram can also be sent to the bus with a time delay.	
Send Output telegram at bus recovery ***Attention! Activate “Behaviour at bus recovery” & set delay in “General settings”.***	**No** Yes
With Yes, the Output of the function block will be sent after bus recovery. This initial Output telegram can also be sent with a delay, which can be configured in “General Settings/Behaviour at bus recovery” – “Delay for sending all status telegrams” If this delay is set, and the behaviour after bus recovery is set to switch the input function block, this switching after bus recovery will not cause an output telegram to be sent to the bus. Only after the initial status delay (as described above) the output telegram will be sent. This delayed sending behaviour is to avoid that all the devices send their output status at the same time after bus recovery (even if all function blocks are switched at the same time after bus recovery) For example, if the delay is set to be 10 seconds and the behaviour after bus return is set to switch the function block ON. Then the output function block will be switched ON immediately after bus recovery (this will not cause any output telegrams to the bus) and then 10 seconds later the output telegrams will be sent.	

5.1.2 Parameter page: FUNCTION BLOCKS / A1…X1/ Advanced Functions

Parameter	Settings
Central ON/OFF function	**No reaction** Any value = ON Any value = OFF 0 = OFF, 1 = ON 0 = ON, 1 = OFF Any value = Timer 1 reaction at ON 0 = X, 1 = ON 0 = OFF, 1 = X
No reaction: the function block output has no reaction when the Central ON/OFF object/s receive/s a telegram. **Any value = ON:** the function block output switches ON when the Central ON/OFF object/s receive/s any telegram (no matter whether “0” or “1” is received). **Any value = OFF:** the function block output switches OFF when the Central ON/OFF object/s receive/s any telegram (no matter whether “0” or “1” is received). **0 = OFF, 1 = ON:** the function block output switches OFF when the Central ON/OFF object/s receive/s a “0” and switches ON when receiving a “1”. **0 = ON, 1 = OFF:** the function block output switches ON when the Central ON/OFF object/s receive/s a “0” and switches OFF when receiving a “1”. **Any value = Timer 1 reaction at ON:** when the Central ON/OFF object/s receive/s any value, the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT ON” will be executed **0 = X, 1 = ON:** the function block output has no reaction when the Central ON/OFF object/s receive/s a “0” and switches ON when receiving a “1”. **0 = OFF, 1 = X:** the function block output switches OFF when the Central ON/OFF object/s receive/s a “0” and has no reaction when receiving a “1”.	
Additional object	**No** Inverted Toggle only with 0 Toggle only with 1 Toggle with 0 and 1
No: this option hides the additional object. **Inverted:** The function block input will invert the value received (ON with a “0” and OFF with a “1”). In other words, it does the opposite to the switching object. **Toggle only with 0:** the function block output will change its state from OFF to ON or vice versa when receiving “0” (it will ignore the telegram when receiving a “1”) **Toggle only with 1:** the ou function block output put will change its state from OFF to ON or vice versa when receiving “1” (it will ignore the telegram when receiving a “0”) **Toggle with 0 and 1:** the function block output will change its state from OFF to ON or vice versa both when receiving “0” or “1”.	
Counters	**No** Yes
There are two counters (one “Run hour” and one “Switching”) per function block available, both of which can be configured to count up or down. **No:** this option hides the counter tab and all its objects and options. **Yes:** this option activates the counter tab.	
Scenes	**No** Yes
KNX standard 1 byte scenes: 1 Scene object per function block. The advantage of having a Scene object per function block (and not only one for the all the function block) is that with the same Scene number, different scenes can be executed (since they are linked to another push button, with a different group address). Up to 8 scenes can be configured per function block. **No:** this option hides the Scenes tab and all scene related functions and object for the current function block. **Yes:** this option activates the Scene tab, with multiple functions and the Scene object for this function block.	

Timer 1 Timer 2	**No** Yes
There are two timers linked to the current function block and which can run parallel; also, they have their own triggering object each. These timers can be configured to works as ON and/or OFF Delay, Staircase, Delay and staircase, blinking, etc. **No**: the Timer tab and all timer related functions are hidden. **Yes:** the Timer tab and the trigger object will be available, but they have no function assigned and this must be configured in the Timer tab.	
Disable	**No** Yes
Each and every function block have a Disable object, which blocks all other functions of the function block. The behaviour at Disabling/Enabling can be configured per function block. **No:** the Disable object and tab will be hidden. **Yes:** this option activates the Disable object and tab.	
Alarms	**No** Yes
Now, in the Advanced Functions of the current function block, you can configure the behaviour of the function block when the alarm objects receive a telegram. After choosing the “Yes” option, the function block -related Alarms tab will be displayed.	

5.1.3 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters

There are two counters (one “Run hour” and one “Switching”) per function block available, both of which can be configured to count up or down.

A) Parameter page: FUNCTION BLOCKS / A1…X1) / Advanced Functions / Counters / Run hour counter

Parameter	Settings
Run hour counter	**No** Upward Backward
No: this option hides the Run hour counter tab and all its objects and options. **Upward:** this option is used to count the accumulated time during which the function block has been switched ON. **Backward:** to count down from a configurable initial value.	

A.1) Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Run hour counter - UP

Parameter	Settings
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**
Usually, a Run hour counter has a 4 bytes unsigned value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values.	
Initial value run hour counter	**No** Yes
Attention! *After programming this value will only be overwritten if the new starting value is changed.*	
This option gives you the possibility to establish an initial value from which the counting will start up. After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter Practical example: should the device be installed in an existing installation, where the load connected to the current function block has already a known number of run-hours, this information can be used as the “New starting value”. But in a later stage, if some other parameter in the actuator must be changed and downloaded, the new current counter value will not be overwritten.	
Run hours threshold value	0
Attention! 0 = Deactivated	

Here you can enter the number of run hours that will trigger the 1 bit alarm object of the current function block. So, this alarm object will be activated and send a “1” to the bus as soon as the Run hour counter passes this threshold. Should the conversion factor be activated and set to be for example “Several run-hours increases 1 step” = 3, and the threshold value is set to 5 then the sequence will be as follows: : 0,0,1,1,1,2,2,2,3,3,3,4,4,4,5,… The alarm is sent in the first 5 after 15 pulses. Attention, this alarm will also be sent to the bus immediately after bus recovery.	
Object for reading / writing the threshold value	**No** Only readable Readable and writable
Only readable: this option will activate an unsigned counter object, which can be read by the ETS/other KNX devices. **Readable and writable:** this option will activate an unsigned counter object, which can be read and overwritten by the ETS/other KNX devices. This is meant to allow changing the threshold value with, for instance, a visualization.	
Reaction on overflow (Max. value of DPT)	**Reset to 0 and start again** Stay at maximum
Attention! Both counter & alarm objects will be set to zero	
Important note: the overflow must not be mistaken with the threshold value, since they are two totally different concepts: An overflow is reached when the object value exceeds the maximum value of the selected data point type. For example, the maximum value of a 1 byte unsigned value is 255; therefore, the overflow is reached when the object value exceeds 255. On the other hand, the threshold refers to any given value of your choice that is valid for this DPT. **Reset to 0 and start again:** when then overflow is reached, the object will start counting from 0 again. Attention! In this case the alarm object will also be set to zero, otherwise one would not know if the threshold has newly been reached or not. **Stay at maximum:** in the event of the overflow being reached, the object will stop at the maximum value of the DPT.	
Additional functions	**No** Yes
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.	

a) Paremeter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Run hour counter – UP / ADDITIONAL FUNCTONS

Parameter	Settings
Cyclic sending of counter value	**No** Yes
When this function is activated, the corresponding object will not send the telegram once, but repeat it infinitely.	
Counter values are sent to the bus every: (Run hours)	1

Enter here the number of hours that must go by before the counter sends its value to the bus. This option is meant to reduce the bus traffic. For instance, if you enter a “5”, the counter will send its first value whenever the accumulated ON time of the function block has reached 5 hours and will then send the value 5 to the bus (10, 15, 20, 25, 30, 35…).	
Conversion factor	**None** Several hours increase 1 step 1 hour increases several steps
None: for each 1 hour accumulated ON time of the function block, the counter increases 1 step. **Several hours increase 1 step:** define here the number of accumulated ON time (in hours) that must go by for the counter to increase 1 step. **1 hour increases several steps:** define here the step increment for each hour of accumulated ON time. For example, after 8 accumulated ON time hours, the counter will have increased 8 x 10 (= 80) steps.	
Send last value of counter at reset by counter object	**No** Yes
No: if you reset the counter by using the 1 bit reset object, the last value of the counter will not be sent to the bus by the counter object. Instead, a “0” will be sent to indicate it has been reset. **Yes:** if you reset the counter by using the 1 bit reset object, the counter object will send its current value before reset to the bus and afterwards it will not reset to 0 but stay at its last value. Only at the next counter step, will the first counter step be sent to the bus. Thus the counter will never have the value “0”.	
Additional object to store last value of counter on reset	**No** Yes Yes and send
No: no additional object to store the last value of the counter on reset will be activated. **Yes:** an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). **Yes and send:** an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.	

A.2) Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Run hour counter - BACK

Parameter	Settings
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**
Usually, a Run hour counter has a 4 bytes unsigned value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values.	
Initial value run hour counter	**8000**
Attention! *After programming this value will only be overwritten is the new starting value is changed.*	

Here you can establish an initial value from which the counter will count back. After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter Introduce here the lifespan of the connected load according to its data sheet which then can be used to supervise the lifespan of a lamp or any given load. It sends an alarm telegram when reaching the value zero. So instead of changing the lamp/load when it fails, it can be done before as a proactive measure. This is especially useful in halls with high ceilings. It cost more for a maintenance callout for changing individual bulbs every time they brake, than making a bulk replacement of all bulbs which or are close to or have reached zero, even though they are still working. Should the conversion factor be activated and set to be for example “Several triggers decreases 1 step” = 3, and the “Initial value switching counter” is set to 5 then the sequence will be as follows: 444,333,222,111,000, and only at the last 0 the alarm will be sent.	
Reaction on reaching zero	Stay at zero **Reset to initial value and start again**
Stay at zero: once the counter reaches 0, it will stay there until it has been reset. **Reset to initial value and start again:** once the counter reaches 0, it will start counting back again starting from the initial value of the run hour counter (as parameterized in the previous option).	
Additional functions	**No** Yes
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.	

a) Paremeter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Run hour counter – BACK / ADDITIONAL FUNCTONS

Parameter	Settings
Cyclic sending of counter value	**No** Yes
When this function is activated, the corresponding object will not send the telegram once, but repeat it infinitely.	
Counter values are sent to the bus every: (Run hours)	1
Enter here the number of hours that must go by before the counter sends its value to the bus. This option is meant to reduce the bus traffic. For instance, if you enter a “5”, the counter will have to count back 5 more hours in order to send the next value to the bus (60, 55, 50, 45, 40…).	
Conversion factor	**None** Several hours decrease 1 step 1 hour decreases several steps
None: for each 1 hour accumulated ON time of the function block, the counter decreases 1 step. **Several hours decrease 1 step:** define here the number of accumulated ON time (in hours) that must go by for the counter to decrease 1 step. **1 hour decrease several steps:** define here the step decrement for each hour of accumulated ON time. For example, after 8 accumulated ON time hours, the counter will have decreased 8 x 10 (= 80) steps.	
Send last value of counter at reset by counter object	**No** Yes
No: if you reset the counter by using the 1 bit reset object, the last value of the counter will not be sent to the bus by the counter object. Instead, a “0” will be sent to indicate it has been reset. **Yes:** if you reset the counter by using the 1 bit reset object, the counter object will send its current value before Reset to the bus and afterwards it will not reset to 0 but stay at its last value. Only at the next counter step, will the first counter step be sent to the bus. Thus the counter will never have the value “0”.	

Additional object to store last value of counter on reset	**No** Yes Yes and send
No: no additional object to store the last value of the counter on reset will be activated. **Yes:** an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). **Yes and send:** an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.	

B) Parameter page: FUNCTION BLOCKS / A1…X1 / ADVANCED FUNCTIONS / Counters / Switching counter

Parameter	Settings
Switching counter	**No** Upward Backward
No: this option hides the Switching counter tab and all its objects and options. **Upward:** this option is used to count the accumulated switching operations of the current function block. **Backward:** to count down from a configurable initial value.	

B.1) Parameter page: FUNCTION BLOCKS / A1…X1 / ADVANCED FUNCTIONS / Counters / Switching counter - UP

Parameter	Settings
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**
Usually, a Switching counter has a 4 bytes unsigned value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values.	
Count number of switching's on:	**Only ON** Only OFF ON and OFF
Only ON: the counter will increase only with ON operations. **Only OFF:** the counter will increase only with OFF operations. **ON and OFF:** the counter will increase with both ON and OFF operations.	
Initial value switching counter	**No** Yes
Attention! *After programming this value will only be overwritten is the new starting value is changed.*	

This option gives you the possibility to establish an initial value from which the counting will start up

After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter

Practical example: should the device be installed in an existing installation, where the load connected to the current function block has already a known number of switching operations, this information can be used as the "New starting value". But in a later stage, if some other parameter in the actuator must be changed and downloaded, the new current counter value will not be overwritten.

Switching threshold value	0

Attention! 0 = Deactivated

Here you can enter the number of switching operations that will trigger the 1 bit alarm object of the current function block. So, this alarm object will be activated and send a "1" to the bus as soon as the switching counter passes this threshold.

Should the conversion factor be activated and set to be for example "Several switching's increases 1 step" = 3, and the threshold value is set to 5 then the sequence will be as follows: : 0,0,1,1,1,2,2,2,3,3,3,4,4,4,5,… The alarm is sent in the first 5 after 15 pulses.

Attention, this alarm will also be sent to the bus immediately after bus recovery.

Object for reading / writing the threshold value	**No** Only readable Readable and writable

Only readable: this option will activate an unsigned counter object, which can be read by the ETS/other KNX devices.
Readable and writable: this option will activate an unsigned counter object, which can be read and overwritten by the ETS/other KNX devices. This is meant to allow changing the threshold value with, for instance, a visualization.

Reaction on overflow (Max. value of DPT)	**Reset to 0 and start again** Stay at maximum

Attention! Both counter & alarm objects will be set to zero

Important note: the overflow must not be mistaken with the threshold value, since they are two totally different concepts:
An overflow is reached when the object value exceeds the maximum value of the selected data point type. For example, the maximum value of a 1 byte unsigned value is 255; therefore, the overflow is reached when the object value exceeds 255.
On the other hand, the threshold refers to any given value of your choice that is valid for this DPT.
Reset to 0 and start again: when then overflow is reached, the object will start counting from 0 again. Attention! In this case the alarm object will also be set to zero, otherwise one would not know if the threshold has newly been reached or not.
Stay at maximum: in the event of the overflow being reached, the object will stop at the maximum value of the DPT.

Additional functions	**No** Yes

In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.

b) Paremeter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Switching counter – UP / ADDITIONAL FUNCTONS

Parameter	Settings
Cyclic sending of counter value	**No** Yes
When this function is activated, the corresponding object will not send the telegram once, but repeat it infinitely.	
Counter values are sent to the bus every: (Switchings)	1
Enter here the number of switching operations that be executed before the counter sends its value to the bus. This option is meant to reduce the bus traffic. For instance, if you enter a “50”, the counter will send its first value whenever the accumulated switching operations of the function block amount to 50 and will then send the value 50 to the bus (50, 100, 150, 200, 250…).	
Conversion factor	**None** Several hours increase 1 step 1 hour increases several steps
None: for each switching operation of the function block, the counter increases 1 step. **Several hours increase 1 step:** define here the number of switching operations that must be executed for the counter to increase 1 step. **1 hour increases several steps:** define here the step increment for each switching operation. For example, after 50 switching operations, the counter will have increased 50 x 10 (= 500) steps.	
Send last value of counter at reset by counter object	**No** Yes
No: if you reset the counter by using the 1 bit reset object, the last value of the counter will not be sent to the bus by the counter object. Instead, a “0” will be sent to indicate it has been reset. **Yes:** if you reset the counter by using the 1 bit reset object, the counter object will send its current value before reset to the bus and afterwards it will not reset to 0 but stay at its last value. Only at the next counter step, will the first counter step be sent to the bus. Thus the counter will never have the value “0”.	
Additional object to store last value of counter on reset	**No** Yes Yes and send
No: no additional object to store the last value of the counter on reset will be activated. **Yes:** an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). **Yes and send:** an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.	

B.2) Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Switching counter - BACK

Parameter	Settings
Data point type of counter	1 byte unsigned 2 bytes unsigned **4 bytes unsigned**

Usually, a Run hour counter has a 4 bytes unsigned value. But 1 and 2 bytes unsigned can also be configured for the purpose of showing the value in info displays, which cannot display 4 bytes unsigned values.	
Count number of switching's on	**Only ON** Only OFF ON and OFF
Only ON: the counter will decrease only with ON operations. **Only OFF:** the counter will decrease only with OFF operations. **ON and OFF:** the counter will decrease with both ON and OFF operations.	
Initial value switching counter *Attention! After programming this value will only be overwritten is the new starting value is changed.*	**8000**
Here you can establish an initial value from which the counter will count back. Attention! This value will never be sent. The 1st value sent will be the first decreased value. It will send a 1 bit alarm telegram with the value "1" when reaching the value zero. After downloading with the ETS this value will only be overwritten if the new starting value is changed. Take into account that the additional counter Introduce here the maximum number of switching's of the connected load, (according to its data sheet) which then can be used to supervise the lifespan of a lamp or any given load. It sends an alarm telegram when reaching the value zero. So instead of changing the lamp/load when it fails, it can be done before as a proactive measure. This is especially useful in halls with high ceilings. It cost more for a maintenance callout for changing individual bulbs every time they brake, than making a bulk replacement of all bulbs which or are close to or have reached zero, even though they are still working. Should the conversion factor be activated and set to be for example "Several triggers decrease 1 step" = 3, and the "Initial value switching counter" is set to 5 then the sequence will be as follows: 444,333,222,111,000, and only at the last 0 the alarm will be sent.	
Reaction on reaching zero	Stay at zero **Reset to initial value and start again**
Stay at zero: once the counter reaches 0, it will stay there until it has been reset. **Reset to initial value and start again:** once the counter reaches 0, it will start counting back again starting from the initial value of the switching counter (as parameterized in the previous option). Attention! This initial value will not be sent to the bus, the next trigger sends the decreased value.	
Additional functions	**No** Yes
In order to keep the application program as easy as possible, only the main and most important functions are displayed at first sight. You will often find the possibility to activate the Additional or Advanced Functions, which disclose new functions that are not essential, but can be very useful.	

b) Paremeter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Counters / Switching counter – BACK / ADDITIONAL FUNCTONS

Parameter	Settings
Cyclic sending of counter value	**No** Yes

When this function is activated, the corresponding object will not send the telegram once, but repeat it infinitely.	
Counter values are sent to the bus every: (Switchings)	1
Enter here the number of switching operations that must be executed before the counter sends its value to the bus. This option is meant to reduce the bus traffic. For instance, if you enter a “50”, the counter will have to count back 50 switching operations in order to send the next value to the bus (550, 500, 450, 400, 350…).	
Conversion factor	**None** Several hours decrease 1 step 1 hour decreases several steps
None: for each 1 switching operation of the function block, the counter decreases 1 step. **Several hours increase 1 step:** define here the number of switching operations that must be executed for the counter to decrease 1 step. **1 hour increases several steps:** de define here the step decrement for each switching operation. For example, after 50 switching operations, the counter will have decreased 50 x 10 (= 500) steps.	
Send last value of counter at reset by counter object	**No** Yes
No: if you reset the counter by using the 1 bit reset object, the last value of the counter will not be sent to the bus by the counter object. Instead, a “0” will be sent to indicate it has been reset. **Yes:** if you reset the counter by using the 1 bit reset object, the counter object will send its current value before reset to the bus and afterwards it will not reset to 0 but stay at its last value. Only at the next counter step, will the first counter step be sent to the bus. Thus the counter will never have the value “0”.	
Additional object to store last value of counter on reset	**No** Yes Yes and send
No: no additional object to store the last value of the counter on reset will be activated. **Yes:** an additional object to store the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). **Yes and send:** an additional object to store and send the last value of the counter on reset will be activated. This object can work parallel with the previous option (Last value of counter at reset by counter object) and it is mainly there to store this last value until the next reset, whereas the counter object only stores it for a short time (until next counter pulse). This value will then be sent after reset using this additional object.	

5.1.4 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Scenes

KNX standard 1 byte scenes: 1 Scene object per output. The advantage of having a Scene object per function block (and not only one for the all the function block) is that with the same Scene number, different scenes can be executed (since they are linked to another push button, with a different group address).

Up to 8 scenes can be configured per function block.

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
Most of the device modules can be deactivated with a "… disable" object. The value (1 or 0) used to disable can also be configured. This option can be very useful for many reasons, including simplifying the configuration: for instance, the logic functions might be a complex task that can take a while to finish; in the meantime, you don't want these modules to be active and cause unwanted actions. Therefore, you can disable them until you finish programming. Another example: you can simply activate/deactivate the timers for the irrigation system when not needed.	

5.1.4.1 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Scenes / Common Scene Parameters

As mentioned before, up to 8 scenes can be configured per function block with identical parameters.

Parameter	Settings
Reaction of function block for	Scene 1 … Scene 64
Attention! Same scene number may not be used twice! Only the first one (top) will prevail	
Here you can define the Scene number where this function block should participate in. All 64 possible KNX scenes can be used. As described in the KNX specifications, in order to reproduce scene 1, the value 0 has to be sent to the scene object of the function block and so on (0=play_scene1 …. 63= play_scene64). Important note: you may not use the same Scene number twice! Should you choose the same Scene number in more than one of the 8 available scene options, only the first one (from top to bottom) will prevail; the other will be ignored.	
Possible to save scene	No **Yes**
It is possible to save the current output state of the actuator as the new scene state. As described in the KNX specifications, in order to save scene 1, the value 128 has to be sent to the scene object of the function block and so on until 192 (128=save_scene1 …. 192= save_scene64). The configured parameter in "Output state for scene" will be overwritten. For example, the end user of the installation can switch ON/OFF the lights as wished and then save the current state for this scene via long press of a standard KNX scene push button. **No:** the scene cannot be saved with the KNX scene object.	

Yes: this option allows to overwrite the current state of the output as the new “Output state for scene”, according to the KNX standardization. **Important note:** if the output state for scene is configured as a “Timer 1 reaction at ON” or “Timer 1 reaction at OFF”, the output state will NOT be saved. The end-user parameters (like this one) can be configured in GENERAL SETTINGS/OVERWRITE END-USER PARAMETER VALUES AT DOWNLOAD. Here you can choose for the “Output state for scene” not to be over-written by ETS download.	
Output state for scene	**No function** ON OFF Timer 1 reaction at ON Timer 1 reaction at OFF
Here you can establish the initial function block state of the scene. Please, note that this can be overwritten by the end user if you have selected “Yes” in the option above (“Possible to save scene”). No function: the function block will have no reaction in the initial stage; the function block will only react to this scene if “save scene” is active and it has been saved by the scene object. ON: the function block switches ON when executing the scene (unless otherwise saved via function block scene object) OFF: the function block switches OFF when executing the scene (unless otherwise saved via function block scene object) Timer 1 reaction at ON: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT ON” will be executed (unless otherwise saved via function block scene object) Timer 1 reaction at OFF: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT OFF” will be executed (unless otherwise saved via function block scene object)	

5.1.5 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2

There are two timers linked to the current function block and which can run parallel; also, they have their own triggering object each. These timers can be configured to works as ON and/or OFF Delay, Staircase, Delay and staircase, blinking, etc.

The Timer trigger object is a 1 bit object which will have different behaviours when receiving an ON or OFF respectively. Next we will explain both REACTION AT ON and REACTION AT OFF separately:

5.1.5.1.1 Parameter page: FUNCTION BLOCKS / Channel A1…X1 / Advanced Functions / Timer 1 and 2 / Reaction at ON

Parameter	Settings
REACTION AT ON	No action Delay **Staircase** Delay and staircase Only ON (without delay/staircase)
The timer can be used as any of the above timer types. These are the possible actions to be executed when the timer trigger object receives an ON (“1”): **No action:** the timer will not be executed. **Delay:** the function block switches ON after a time delay. **Staircase:** the function block immediately switches ON and stays ON for the configured staircase time and thereafter switches OFF again. **Delay and staircase:** the function block switches ON after a time delay and then stays ON for the configured staircase time and thereafter switches OFF again. **Only ON (without delay/staircase):** the function block immediately switches ON and stays ON.	

Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2 / Reaction at ON / Delay

Parameter	Settings

- ON delay Base	1 s
- ON delay Factor	10
Configure here the time delay for the function block to switch ON	

Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2 / Reaction at ON / Staircase

Parameter	Settings
- Staircase time (ON duration) Base	**1 s** 5 s 10 s 1 min 5 min 10 min 1 h
- Staircase time (ON duration) Factor	**60**
Establish here the wished time for the function block to be ON The Staircase time is the period of time during which the device function block will be switched ON. After this time elapses, the function block switches OFF again.	
- Factor changeable by object / Remaining time cyclic sending	**No** Yes
No (default option): staircase time only configurable via parameters. Yes: this option activates an object to change staircase time factor. As you can see in the picture below, the time Base can be any of the following: So, if you have selected, for instance, “1 s”, then the values received in this object will be in “seconds”. If you have selected “5 s” though, the values received will be in “seconds” and multiplied by 5 (base “5 s” x value received at object “10” = “50 seconds”). The same rule applies if the Base has been selected in “minutes” or “hours”. When using this communication object to modify the staircase factor, if the modification is done while the staircase is active , the modification will be applied after the end of the current staircase Additionally, to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value until the timer finish. In order to disable this function, the “T” flag must be deactivated.	
Advanced staircase function	**No** Yes
Here the advanced functions can be activated.	

Parameter page: FUNCTION BLOCKS / A1…X1 (Binary) / Advanced Functions / Timer 1 and 2 / Reaction at ON / ADVANCED STAIRCASE FUNCTIONS

Parameter	Settings
Multiply staircase	**No** Yes

* With Yes: ***Attention! Total staircase time = staircase time x number of consecutive ON telegrams separated by less than 1 sec. from each other***

Here you can activate the possibility to multiply the staircase time in order to extend the time during which the function block will stay ON. The total staircase ON time is calculated by taking the parameterized staircase time and multiplying it by the number of ON telegrams received.

This resulting time will never exceed the parameterized maximum staircase in the option “Maximum staircase time Base/Factor”

It is important to keep in mind that the multiplication will only be done starting from the first triggering telegram (so, the Multiplying staircase function will only be executed when starting the staircase, not during execution). Therefore, these ON telegrams may not be longer than 1 second apart. Should more than 1 second elapse between two telegrams, then it will only do the multiplication of the previous pulses received. The telegrams received after this, will be ignored or interpreted as a retrigger timer function (if parameterized).

Practical example: as implied by its name, the staircase time is frequently used in staircases. With the purpose of lowering the costs, instead of using a movement detector for switching ON/OFF, often push buttons are used with the staircase time as defined in the actuator. In order to save energy, the staircase time should be as short as possible, but sometimes you may wish to have the lights longer ON. In this case, this option can be very useful because it allows the end user to easily extend the staircase time by pressing several times (depending on how long the light should stay ON).

Retrigger timer	No **Yes, excluding multiplication** Yes, including multiplication

It is possible to extend the staircase time by retriggering it (in other words, the timer starts counting again from the start). But this function will only be executed after more than 1 second has elapsed between the triggering events of the timer (if less than 1 second, see behaviour in section MULTIPLY STAIRCASE).

No: the staircase will not be retriggered.

Yes, excluding multiplication (default option): this option will retrigger the staircase to be reset to the time (Base/Factor) as configured in the ETS application program.

For example: you have configured the staircase time in the ETS application program to be 1 minute; should the staircase time be, for instance, 1 hour as the result of a previous multiplication (Multiply staircase option), the moment you receive the retrigger telegram it will be reset to 1 minute again.

Yes, including multiplication: this option will retrigger the staircase to be reset to the current staircase time (it could be the parameterized time or the multiplied staircase time).

For example: you have configure the staircase time in the ETS application program to be 1 minute; should the staircase time be, for instance, 1 hour as the result of a previous multiplication (Multiply staircase option), the moment you receive the retrigger telegram it will be reset to 1 hour again.

Warning pulse	**No function** With own output With additional object

The warning pulse is meant to inform the end user about the fact that the staircase time is about to expire.

No function: the light will go OFF without previous warning after the staircase time elapses.

With own output: the same function block will be used for this warning pulse.

The function block, according to the default parameters, the function block output will switch OFF 10 seconds before the end of the staircase time and it will switch ON again 2 seconds after switching OFF. This creates a short blinking effect as a visual warning.

It is important to be able to configure the OFF time because not all loads can switch OFF immediately (for example, lights using transformers). So, if you have selected 1 second as a warning time, it might not switch OFF at all.

With additional object: this option serves the same purpose of warning before the staircase time elapses. It is specially indicated for those places where the function block can/may not be switched ON and OFF quickly. In these cases, the additional object can send a warning pulse to another function block (different load) just before the end of the staircase time of the main load.

Practical example: let's say this function block is used to control the flood lights of a tennis court via contactor. These lights take long to switch ON again (after they have been switched OFF), which is not energy-efficient nor practical. Therefore, to be able to generate a warning pulse, you can use an additional warning light connected to another function block, which this additional object is linked to.

1 action: ON: the additional object only sends a "1" at the configured point in time before the staircase time elapses.

2 actions: 1st OFF, 2nd ON: the additional object can execute two actions by sending:
Time before end of staircase for 1st action: a "0" at the configured point in time before the staircase time elapses.
Time before end of staircase for 2nd action: a "1" at the configured point in time before the staircase time elapses.

2 actions : 1st ON, 2nd OFF: the additional object can execute two actions by sending:

Time before end of staircase for 1st action: a "1" at the configured point in time before the staircase time elapses.
Time before end of staircase for 2nd action: a "0" at the configured point in time before the staircase time elapses.

3 actions: 1st OFF, 2nd ON, 3rd OFF (default option): the additional object can execute three actions by sending:
Time before end of staircase for 1st action: a "0" at the configured point in time before the staircase time elapses.
Time before end of staircase for 2nd action: a "1" at the configured point in time before the staircase time elapses.
Time before end of staircase for 3rd action: a "0" at the configured point in time before the staircase time elapses.

Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2 / Reaction at ON / Delay and staircase

The Staircase function has been explained above. This "Delay and Staircase" combined function could also have:

Parameter	Settings
- ON delay Base	**1 s**
- ON delay Factor	**10 s**
The staircase can start after a configurable time delay	

- Staircase time (ON duration) Base	**1 s**
- Staircase time (ON duration) Factor	**60 s**
Establish here the wished time for the function block to be ON The Staircase time is the period of time during which the device function block will be switched ON. After this time elapses, the function block switches OFF again.	
- Factor changeable by object / Remaining time cyclic sending	**No** Yes
No (default option): staircase time only configurable via parameters. **Yes:** this option activates an object to change staircase time factor. As you can see in the picture below, the time Base can be any of the following: So, if you have selected, for instance, “1 s”, then the values received in this object will be in “seconds”. If you have selected “5 s” though, the values received will be in “seconds” and multiplied by 5 (base “5 s” x value received at object “10” = “50 seconds”). The same rule applies if the Base has been selected in “minutes” or “hours”. Additionally, to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value until the timer finish. In order to disable this function, the “T” flag must be deactivated.	
Blinking / number of repetitions (0 = none, 65535 = infinite)	**0**
A repeated staircase function with an initial delay actually becomes a blinking function. It is indicated to switch a load ON and OFF with a configurable certain frequency (which can have different ON and OFF times). The number of repetitions can be configured and can also be set to any number between 1 and 65534.A. Infinite repetitions can be achieved by using the value 65535. In order to deactivate the blinking, just enter the value 0.	

5.1.5.1.2 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2 / Reaction at OFF

Parameter	Settings
REACTION AT OFF	No action **OFF without delay** OFF with delay
Attention! Reaction at OFF cancels the running staircase	
This are the possible actions to be executed when the timer trigger object receives an OFF (“0”): **No action:** the timer will not be interrupted. **OFF without delay:** the function block immediately switches OFF and the timer function is cancelled.	

OFF with delay: the function block switches OFF after a time delay.
As soon as the OFF telegram is received, the Timer is cancelled.

Object to disable timer	Yes, immediately Yes, on ending current timer **No**

The disable object will always react as follows (and cannot be otherwise configured):

"1": disable.
"0": enable.

Yes, immediately: as soon as the Disable object receives a "1", the timer will be cancelled and disabled. This option activates the parameter "Reaction on bus voltage recovery".

Yes, on ending current timer: whenever the Disable object receives a "1", the timer will be not cancelled, but disabled. Thus, the current timer will finalize normally. This option activates the parameter "Reaction on bus voltage recovery".

No: the disable object, including the "Reaction on bus voltage recovery" will be hidden.

A) Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Timer 1 and 2 /
Reaction at OFF / Object to disable timer

Parameter	Settings
Object to disable timer	**Yes, immediately** Yes, on ending current timer No

The disable object will always react as follows (and cannot be otherwise configured):
"1": disable.
"0": enable.
Yes, immediately: as soon as the Disable object receives a "1", the timer will be cancelled and disabled. This option activates the parameter "Reaction on bus voltage recovery".
Yes, on ending current timer: whenever the Disable object receives a "1", the timer will be not cancelled, but disabled. Thus, the current timer will finalize normally. This option activates the parameter "Reaction on bus voltage recovery".
No: the disable object, including the "Reaction on bus voltage recovery" will be hidden.

A.1) Parameter page: FUNCTIONAL BLOCKS / A1…X1 / ADVANCED FUNCTIONS / Timer 1 and 2 /
Reaction at OFF / Object to disable timer / Reaction on bus voltage recovery

Parameter	Settings
Reaction on bus voltage recovery	**Enable** Disable Last object status

Whether the Timer will be active or not on bus voltage recovery can be configured here.

On bus voltage recovery the timer can be enabled, disabled, or have the same state as before the bus failure depending on the above selection.

Enable: the timer will be enabled.
Disable: the timer will be disabled.
Last object status: the status of the Enable object will be saved in the actuator's non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will set the object as it was before the bus failure.

B) Parameter page: FUNCTIONAL BLOCKS / A1…X1 / ADVANCED FUNCTIONS / Timer 1 and 2 / Reaction at OFF / Reaction when SWITCHING or SCENE objects receive a value while timer is active

Parameter	Settings
Reaction when SWITCHING or SCENE objects receive a value while timer is active	Don't cancel timer and do action **Cancel timer and do action** Ignore telegram
Don't cancel timer and do action: the Switching or Scene function will not cancel the active timer and the function will be executed parallel to the Timer. **Cancel timer and do action:** the Switching or Scene function will cancel the active timer and only the triggered functions (Switching or Scene) will be executed (whereas the Timer will be cancelled and thus will not interfere with these functions). **Ignore telegram:** if a telegram is received via the Switching or Scene objects while the timer is active, these functions (Switching or Scene) will not be executed.	

5.1.6 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Disable

Each and every function block has a Disable object, which blocks all other functions of the function block.

The behaviour at Disabling/Enabling can be configured per function block.

On the other hand, the priority of all Disable objects can also be adjusted to have higher/lower priority as the alarms; this can be done in General Settings/Advanced Functions/Alarms (then, Alarm tab)

Parameter	Settings
Disable object	**Disable with ON** Disable with OFF
Disable with ON: the function block will be blocked whenever the Disable object receives a “1”; and enabled again with a “0”. **Disable with OFF:** the function block will be blocked whenever the Disable object receives a “0”; and enabled again with a “1”.	
- Reaction on bus voltage recovery	**Enable** Disable Last object status
Whether the function block will be disabled or enabled on bus voltage recovery can be configured here. **Enable:** the function block will be enabled. **Disable:** the function block will be disabled. **Last object status:** the status of the Enable object will be saved in the actuator’s non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will set the object as it was before the bus failure.	
Behaviour at disabling	**Block function block as is** ON OFF Timer 1 reaction at ON Timer 1 reaction at OFF
Block function block as is: the function block will be blocked, but not switched ON or OFF when disabling the function block via Disable object. **ON:** the function block will be switched ON and blocked. **OFF:** the function block will be switched OFF and blocked. Each output has two timer functions. Only the first timer can be assigned to the behaviour at disabling: **Timer 1 reaction at ON:** the function that has been chosen under “FUNCTION BLOCKS/Timer 1/REACTION AT ON” will be executed and the function block will be blocked. **Timer 1 reaction at OFF:** the function that has been chosen under “FUNCTION BLOCKS/Timer 1/REACTION AT OFF” will be executed and the function block will be blocked.	
Behaviour at enabling	**Enable and leave function block as is** ON OFF Timer 1 reaction at ON Timer 1 reaction at OFF Set to tracked state
Enable and leave function block as is: the function block will be enabled, but not switched ON or OFF when enabling the function block via Disable object. **ON:** the function block will be switched ON and enabled. **OFF:** the function block will be switched OFF and enabled. Each function block output has two timer functions. Only the first timer can be assigned to the behaviour at enabling:	

Timer 1 reaction at ON: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT ON” will be executed and the function block will be enabled.
Timer 1 reaction at OFF: the function that has been chosen under “FUNCTION BLOCK/Timer 1/REACTION AT OFF” will be executed and the function block will be enabled.

Set to tracked state: while the function block is blocked, the other function block -related objects might receive telegrams. Nevertheless, since the function block is blocked, it does not switch ONº or OFF.

Even though the actuator does not switch ON or OFF, it does register all these events in order to be able to go to the state where it would have been at enabling (if the function block had not been blocked).

Attention! Enable function block will trigger the behaviour of the next active (lower priority) alarm. Also the “Behaviour at enabling” will only be executed with no active & acknowledged function block alarms.

5.1.7 Parameter page: FUNCTION BLOCKS / A1…X1 / Advanced Functions / Alarms

Attention! Alarm function must be activated in "General Settings" tab

First of all, in order for the function block-related Alarms to work, the Alarms must be activated in "General Settings/Advanced Functions/Alarms". In this tab you can configure up to 8 alarms to be either "analogue" or "digital".

Function block-dependent alarms: now, in the Advanced Functions of the current function block, you can configure the behaviour of the function blockwhen the alarm objects receive a telegram.

After choosing the "Yes" option, the function block-related Alarms tab will be displayed.

Alarm telegrams are used to block the function block. The reaction of the current function block when any/several of the 8 available alarms have been activated can be configured in the next tab.

Parameter	Settings
Behaviour at beginning of alarm 1…8	**Nothing** Block function block as is ON OFF Timer 1 reaction at ON Timer 1 reaction at OFF
Nothing: the function block will not participate in the alarm. Thus, it will not be blocked. **Block function block as is:** the function block will be blocked, but not switched ON or OFF when activating the alarm. **ON:** the function block will be switched ON and blocked. **OFF:** the function block will be switched OFF and blocked. Each output has two timer functions. Only the first timer can be assigned to the behaviour of the alarm: **Timer 1 reaction at ON:** the function that has been chosen under "FUNCTION BLOCKS/Timer 1/REACTION AT ON" will be executed and the function block will be blocked. **Timer 1 reaction at OFF:** the function that has been chosen under "FUNCTION BLOCKS/Timer 1/REACTION AT OFF" will be executed and the function block will be blocked.	
Behaviour at end of all alarms	**Nothing** ON OFF Timer 1 reaction at ON Timer 1 reaction at OFF Set to tracked state
Attention! *The "Behaviour at end of all alarms" will only be executed with no active & acknowledged function block alarms, and if the "disable function block function" is in enabled state. Only then, the function block will be unblocked.*	

Here you can define the behaviour of the current function block when no alarm is active anymore.

Important note: in the General Settings tab you can configure whether or not the alarms must be acknowledged. The “Behaviour at end of all alarms” will only be executed with no active & acknowledged function block alarms, and if the “disable function block function” is in enabled state. Only then, the function block will be unblocked.

Nothing: the function block will not do anything when enabled.
ON: the function block will be switched ON when enabled.
OFF: the function block will be switched OFF when enabled.
Each output has two timer functions. Only the first timer can be assigned to the behaviour at enabling:
Timer 1 reaction at ON: the function that has been chosen under “FUNCTION BLOCKS/Timer 1/REACTION AT ON” will be executed when enabled.
Timer 1 reaction at OFF: the function that has been chosen under “FUNCTION BLOCKS/Timer 1/REACTION AT OFF” will be executed when enabled.

Set to tracked state: while the function block is blocked, the other function block -related objects might receive telegrams. Nevertheless, since the function block is blocked, it does not switch ON or OFF.
Even though the actuator does not switch ON or OFF, it does register all these events in order to be able to go to the state where it would have been at enabling (if the function block had not been blocked).

5.2 Parameter page: Alarms

Parameter	Settings
Alarms	**No** Yes

First of all, in order for the function block-related Alarms to work, the Alarms must be activated by selecting yes.

Then up to 8 alarms to be either “analog” or “digital” can configured

Now, in the Advanced Functions of the inputs-dependent alarms which can be found in FUNCTIONAL BLOCK/AX/Advanced functions/Alarms, you can configure the behaviour of the function blocks when the alarm objects receive a telegram.

Alarm telegrams are used to block the function block. The reaction of the current function block when any/several of the 8 available alarms have been activated can be configured in the Alarms tab in the output.

Terminology for alarms:
Alarm X enabled / disabled: The alarm can be disabled with the “Alarm X disable” object. This leaves the alarm without any function.

Alarm active / Alarm activated: This means that the alarm has receive a telegram on its “Alarm X” object which triggers the alarm in its active state. This causes the function blocks (depending on the function blocks parameters) to be blocked.

Alarm is triggered: if the alarm is activated while it was already active it will not be triggered if “only the first time” is selected in the trigger parameter.

Alarm inactive / Alarm deactivated / Alarm not active / Alarm ended: This means that the alarm has receive a telegram on its “Alarm X” object which ends the alarm in its inactive state.

Function block disabled: Each function block has a “[X] Disable function blocks” object with which the function block can be blocked.

Function block enabled: Each function block has a “[X] Disable function block” object with which the function block can be enabled. It will only be unblocked though with no active and acknowledged function block alarms

Function block blocked: Due to an active alarm or if the function block was disabled with the “[X] Disable function block” object the function block will be blocked.

Function block unblocked: The function block will only be unblocked with no active and acknowledged function block alarms and if the “disable function block function” is in the enabled state.

Alarm acknowledged: An alarm can only be acknowledged if it is not active. If the acknowledge function is active the function block will have no reaction (no change in the output nor can it be unblocked) until the alarm is acknowledged. This is independent of the “disable function block object” i.e. the alarm can be acknowledged even though the function block is disabled.

Example Alarms Table with “Acknowledge needed” active, and “Priority of disable object for all function block ” > Alarm 2.

This table describes the different behaviours (on the right of the grey column) with consecutive events (left side of the grey column) The order of the events and their respective behaviours are indicated by a number staring for the first event/behaviour with 1 and counting up with each new event. For example, line two:

Event (left side of the grey column)	Behaviour (on the right of the grey column)
1) Alarm 1 is activated	1) Behaviour alarm 1 & Block function block
2) An acknowledge is received	2) No reaction
3) Alarm 1 is deactivated	3) No reaction
4) An acknowledge is received	4) Behaviour at end of all alarms & Unblock Channel

Alarm 1 = 0	Alarm 1 = 1	Disable	Enable	Alarm 2 = 0	Alarm 2 = 1	Ack		Behaviour alarm 1	Behaviour at disable	Behaviour at enable	Behaviour alarm 2	Behaviour at end of all alarms	Block function block	Unblock Channel	No reaction	Alarms ACK but do Nothing
						1									1	
3	1					2, 4		1				4	1	4	2, 3	
2	1					3		1				3	1	3	2	
		1	2						1	2			1	2		
				2	1	3					1	3	1	3	2	
3.1	1	2	4			3.2, 5		1	3.2	4			1	4	2	
3	1	2	4			5		1		4		5	1	5	2, 3, 4	
3.1	1			4	2	3.2, 5		1			3.2	5	1	5	2, 3.1, 4	
3	2	1	5			4		2	1, 4	5			1	5	3	
		2	5	3	1	4			2	5	1		1	5	3	4
		2	4	3	1	5			2		1	5	1	5	3, 4	
6	3	2	5	4	1	7		3	2		1	7	1	7	4, 5, 6	
5	3	2	7	4	1	6		3	2, 6	7	1		1	7	4, 5	6
		2	3	4	1	5			2		1, 3	5	1	5	4	
4.1	3	2	5	6	1	4.2, 7		3	2, 4.2		1, 5	7	1	7	6, 4.1	
3	1	2	5			4		1	4	5			1	5	2, 3	
		2	4	3	1			1	2		4?		1		3, 4?	

Parameter	Settings
Alarm 1	No **Yes**
By default, the first alarm is activated. This option activates or hides the alarm tab with all its parameters.	
Alarm 2…8	**No** Yes
By default, the first alarm is deactivated. This option activates or hides the alarm tab with all its parameters.	
Acknowledge needed	Ack. with 0 Ack. with 1 **No**
* Ack. with 0 / 1: ***Attention! Acknowledge will not execute the "Behaviour at end of all alarms" if the "disable function block object" is in disabled state, but if all alarms have ended, they will be acknowledged.***	
By activating this function, the alarm must be acknowledged (either with a 1 or with a 0 depending on the above parameter selection) in order to unblock the function block. An alarm can only be acknowledged if it is not active. The function block will have no reaction (no change in the output nor can it be unblocked) until the alarm is acknowledged. This is independent of the "disable function block object" i.e. the alarm can be acknowledged even though the function block is disabled.	
Priority of disable object for all function blocks	**< Alarm 8** > Alarm 1 > Alarm 2 > Alarm 3 > Alarm 4 > Alarm 5 > Alarm 6 > Alarm 7 > Alarm 8
Each and every function block has a Disable object, which blocks all other functions of the function block. The behaviour at Disabling/Enabling can be configured per function block. The priority of all Disable objects can here be adjusted to have higher/lower priority as the alarms.	

5.2.1 Parameter page: Alarm 1…8

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	
Type of alarm	**Digital** Analog
Both digital and analog alarms can be used.	

5.2.2 Parameter page: Alarms / Digital

Parameter	Settings

Digital alarm is active when receiving	**On** Off
This parameter is to decide with which useful data of the telegram the alarm will be activated.	
Object to disable Alarm	**No** Yes
The alarm can be disabled with a one bit object. It will be disabled with a 1 and enabled with a 0	
Reaction on bus voltage recovery	**Enable** Disable Last object status
On bus voltage recovery the alarm can be enabled, disabled, or have the same state as before the bus failure depending on the above selection.	
Monitoring time base	**10 s** 1 min 5 min 10 min 1 h
The alarm object must receive a telegram within this time, otherwise the alarm will become active.	
Alarm is triggered	**Always** Only first time
This parameter indicates if the alarm should be triggered each time it is activated or if it should only be triggered the first time. If the alarm is activated while it was already active it will not be triggered if “only the first time” is selected.	

5.2.3 Parameter page: Alarms / Analog

Parameter	Settings
Input value Analog alarm	1 byte unsigned 1 byte scaling **2 bytes float** 4 bytes unsigned 4 bytes float
The analog alarms can have any of the above datapoint types. With the analog alarms you only need to have sensors to send the analog values. You are not forced to use the usually very “rigged” logic of a KNX whether station. Apart from not being flexible to create the correct condition one only disposes of the number of threshold of the weather station. On the other hand with this function in the actuator there are much more thresholds.	
Alarm setpoint [x 0.1]	**300**
This is the setpoint of the analog alarm.	
Hysteresis [x 0.1]	**10**
This is the hysteresis of the analog alarm	
Type of Hysteresis (Threshold calculation)	**Setpoint = Upper Threshold** Setpoint = Lower Threshold Setpoint = Symmetric (1/2 between THs)

The hysteresis can be asymmetric or symmetric as can be seen in the above options. If Setpoint = Upper Threshold, then the Lower Threshold = Setpoint – Hysteresis If Setpoint = Lower Threshold, then the Upper Threshold = Setpoint + Hysteresis If Setpoint = Symmetric (1/2 between THs) then the Upper Threshold = Setpoint + ½ Hysteresis and the Lower Threshold = Setpoint - ½ Hysteresis	
Objects for changing Setpoint/Hysteresis values	**No** Yes
* With Yes ***Attention! The end-user parameter values will only be maintained when "Overwrite end-user…" in general tab were set to "Don't overwrite".***	
Both the setpoint value and the Hysteresis can be changed from the bus. Together with a visualization the customer can adjust each and every threshold to his own criteria. E.g. Wind speed for the awnings, light lux level for the blind position, sun position to move the slats of the blinds, etc.	
Analog alarm is active when	**Exceeding/equal upper threshold** Falling below/equal lower threshold Between upper and lower threshold >/= upper or </= lower threshold
This is to decide when the analog alarm should be active and when it should end (be inactive).	
Object to disable alarm	No **Yes**
The alarm can be disabled with the "Alarm X disable" object. This leaves the alarm without any function.	
Reaction on bus voltage recovery	**Enable** Disable Last object status
On bus voltage recovery the alarm can be enabled, disabled, or have the same state as before the bus failure depending on the above selection.	
Monitoring time base	**10 s** 1 min 5 min 10 min 1 h
The alarm object must receive a telegram within this time, otherwise the alarm will become active.	
Alarm is triggered	Always **Only first time**
This parameter indicates if the alarm should be triggered each time it is activated or if it should only be triggered the first time. If the alarm is activated while it was already active it will not be triggered if "only the first time" is selected.	

5.3 Parameter page: Logics

There are 20 logic functions available

Parameter	Settings
Logics	**No** Yes
The logic functions can be activated here.	

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	
Type of logic	No function **Boolean** Gate / Filter Mathematical Comparators Converters
One of the above logic functions can be selected.	

5.3.1 Parameter page: Logics / Boolean

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Type of Boolean function	**AND** NAND OR NOR XOR XNOR
One of the following Boolean logic functions can be configured.	

5.3.1.1 Parameter page: Logics / Boolean / Input

Parameter	Settings

Input 1 Input 2	**Yes** Yes, inverted
The inputs can be activated or inverted	
Input 3 Input 4	**No** Yes Yes, inverted
The inputs can be activated, deactivated or inverted	
Reaction with event on input	**Execute logic** Don't execute logic
The logic can be executed (triggered) with an event on the input or not depending on the above selection. If "Don't execute logic" is selected the input will change and will not execute the logic, but if another input receives a value it will take the received value into account.	
Input constant / value after bus recovery	**Value before bus failure** Read on init after initial delay Set input to 0 Set input to 1
The input can be set to a constant value by the parameter "set input to X" given it is not changed from the bus afterwards It can also read the value from the bus after bus recovery, or be saved on bus failure in order to set this value on bus voltage recovery. When it is set to read the value after bus recovery, and in the output of the logic "Execute on init." is set to "Yes", then the answers of the read requests will not execute the logic. (unless the delay of the read requests is set to be greater than 2 seconds) The output will be sent with the reaction of the "Execute on init." command.	

5.3.1.2 Parameter page: Logics / Boolean / Output

Parameter	Settings
Datapoint type of output	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Sending condition	**On change** Always
In this parameter one can decide when the value must be sent. If the value must change in order to send it or not.	
Send when true	No **Yes**
If a value should be sent when true	
Value when true	1
Set here the value that should be sent when true	

Send when false	No **Yes**
If a value should be sent when false	
Value when false	0
Set here the value that should be sent when false	
Cyclic sending time	**No** Send when true Send when false Both
If a value should be sent cyclically when true, false or both.	
Execute on init	**No** Yes
The function will be executed after bus voltage recovery if “yes” is selected. With “No”: Attention! If No is selected, not even the response of the read on init will execute the logic With “Yes” and the inputs set to read on init, the output is calculated with all response telegrams	

5.3.2 Parameter page: Logics / Gate/Filter

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Reaction on bus voltage recovery of both disable objects	**Enable** Disable Last object status
On bus voltage recovery the logic can be enabled, disabled, or have the same state as before the bus failure depending on the above selection.	

5.3.2.1 Parameter page: Logics / Gate/Filter / Input

Parameter	Settings
Datapoint type	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float

	4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Reaction of output with event on input	**Always** On change Don't send telegram
The reaction of output with event on input can be configured with the above options	
Enable / Disable GATE/FILTER	No **En = 1 / Dis = 0** En = 0 / Dis = 1
This is the enable / disable input of the gate (not of the logic block) Depending of the above selection the gate will let the values of the input through to the output or not.	
Trigger input to output on en-/disable	**Nothing** Always, on every enable telegram Only when changed from disabled to enabled Always, on every disable telegram Only when changed from enabled to disabled Always, on every en-/disable telegram
The input will be triggered to the output when receiving a telegram on the Enable / disable input independent of the in/out sending conditions. One can decide with this parameter when to do the trigger.	
Input constant / value after bus recovery	**Value before bus failure** Read on init after initial delay Set input to value
The input can be set to a constant value by the parameter "set input to value" given it is not changed from the bus afterwards It can also read the value from the bus after bus recovery, or be saved on bus failure in order to set this value on bus voltage recovery.	

5.3.2.2 Parameter page: Logics / Gate/Filter / Output

Parameter	Settings
Datapoint type of output	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float

For this function one of the above standard KNX datapoint types can be selected.	
Sending condition	**On change** Always
In this parameter one can decide when the value must be sent. If the value must change in order to send it or not.	
Cyclic sending	**No** Yes
The telegram will be repeated cyclically (with a configurable frequency)	
Output filter	**No** Only let through within range Only let through outside of range
The values to be let through or not (filtered) can be configured here.	
Execute on init	**No** Yes
The function will be executed after bus voltage recovery if “yes” is selected. With “No”: Attention! If No is selected, not even the response of the read on init will execute the logic With “Yes” and the inputs set to read on init, the output is calculated with all response telegrams	

5.3.3 Parameter page: Logics / Mathematical

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	

Type of mathematical function	**ADD** SUBSTRACT MULTIPLY DIVIDE MAXIMUM MINIMUM AVERAGE
The type of mathematical function can be selected from one of the options above.	

5.3.3.1 Parameter page: Logics / Mathematical / Input

Parameter	Settings
Input 1 Input 2	No **Yes**
The inputs can be activated or inverted	
Input 3 Input 4	**No** Yes
The inputs can be activated, deactivated or inverted	
Datapoint type of input	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Reaction with event on input	**Execute logic** Don’t execute logic
The logic can be executed (triggered) with an event on the input or not depending on the above selection. If “Don’t execute logic” is selected the input will change and will not execute the logic, but if another input receives a value it will take the received value into account.	
Input constant / value after bus recovery	**Value before bus failure** Read on init after initial delay Set input to value
The input can be set to a constant value by the parameter “set input to value” given it is not changed from the bus afterwards It can also read the value from the bus after bus recovery, or be saved on bus failure in order to set this value on bus voltage recovery.	

5.3.3.2 Parameter page: Logics / Mathematical / Output

Parameter	Settings
Datapoint type of output	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed

	2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Sending condition	**On change** Always
In this parameter one can decide when the value must be sent. If the value must change in order to send it or not.	
Cyclic sending	**No** Yes
The telegram will be repeated cyclically (with a configurable frequency)	
Output filter	**No** Only let through within range Only let through outside of range
The values to be let through or not (filtered) can be configured here.	
Execute on init	**No** Yes
The function will be executed after bus voltage recovery if “yes” is selected. With “No”: Attention! If No is selected, not even the response of the read on init will execute the logic With “Yes” and the inputs set to read on init, the output is calculated with all response telegrams	

5.3.4 Parameter page: Logics / Comparators

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Type of comparators function	**EQUAL** GREATER SMALLER GREATER OR EQUAL SMALLER OR EQUAL DISTINCT
The type of comparator function can be selected from one of the options above.	

5.3.4.1 Parameter page: Logics / Comparators / Input

Parameter	Settings
Input 1 Input 2	No **Yes**
The inputs can be activated or inverted	
Input 3 Input 4	**No** Yes
The inputs can be activated, deactivated or inverted	
Datapoint type of input	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Reaction with event on input	**Execute logic** Don’t execute logic
The logic can be executed (triggered) with an event on the input or not depending on the above selection. If “Don’t execute logic” is selected the input will change and will not execute the logic, but if another input receives a value it will take the received value into account.	
Input constant / value after bus recovery	**Value before bus failure** Read on init after initial delay Set input to value
The input can be set to a constant value by the parameter “set input to value” given it is not changed from the bus afterwards It can also read the value from the bus after bus recovery, or be saved on bus failure in order to set this value on bus voltage recovery.	

5.3.4.2 Parameter page: Logics / Comparators / Output

Parameter	Settings
Datapoint type of output	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Sending condition	**On change** Always

In this parameter one can decide when the value must be sent. If the value must change in order to send it or not.	
Send when true	No **Yes**
If a value should be sent when true	
Value when true	1
Set here the value that should be sent when true	
Send when false	No **Yes**
If a value should be sent when false	
Value when false	0
Set here the value that should be sent when false	
Cyclic sending time	**No** Send when true Send when false Both
If a value should be sent cyclically when true, false or both.	
Execute on init	**No** Yes
The function will be executed after bus voltage recovery if “yes” is selected. With “No”: Attention! If No is selected, not even the response of the read on init will execute the logic With “Yes” and the inputs set to read on init, the output is calculated with all response telegrams	

5.3.5 Parameter page: Logics / Converters

Parameter	Settings
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	

5.3.5.1 Parameter page: Logics / Converters / Input

Parameter	Settings

Datapoint type of input	**1 bit** 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Reaction with event on input	**Execute logic** Don't execute logic
The logic can be executed (triggered) with an event on the input or not depending on the above selection. If "Don't execute logic" is selected the input will change and will not execute the logic, but if another input receives a value it will take the received value into account.	
Input constant / value after bus recovery	**Value before bus failure** Read on init after initial delay Set input to value
The input can be set to a constant value by the parameter "set input to value" given it is not changed from the bus afterwards It can also read the value from the bus after bus recovery, or be saved on bus failure in order to set this value on bus voltage recovery.	

5.3.5.2 Parameter page: Logics / Converters / Output

Parameter	Settings
Datapoint type of output	1 bit 1 byte scaling **1 byte unsigned** 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
For this function one of the above standard KNX datapoint types can be selected.	
Sending condition	**On change** Always
In this parameter one can decide when the value must be sent. If the value must change in order to send it or not.	
Cyclic sending	**No** Yes

The telegram will be repeated cyclically (with a configurable frequency)	
When result value exceeds max. allowed DPT of output value:	Don't send **Send max. value of output** Send value
An overflow is reached when the object value exceeds the maximum value of the selected data point type. For example, the maximum value of a 1 byte unsigned value is 255; therefore, the overflow is reached when the object value exceeds 255. If the result exceeds this maximum DPT value one can select to not send anything, send max. value of output, or send a predefined value.	
When result value is lower than allowed DPT of output value:	Don't send **Send min. value of output** Send absolute value (without sign) Send value
If the result is lower than the minimum value of the DPT one can select to not send anything, send min. value of output, Send absolute value (without sign) or send a predefined value.	
Output filter	**No** Only let through within range Only let through outside of range
The values to be let through or not (filtered) can be configured here.	
Execute on init	**No** Yes
The function will be executed after bus voltage recovery if "yes" is selected. With "No": Attention! If No is selected, not even the response of the read on init will execute the logic With "Yes" and the inputs set to read on init, the output is calculated with all response telegrams	

5.4 Parameter page: Scene controller

Parameter	Settings
Scene controller	**No** Yes
The actuator can also be used as a scene controller with a KNX scene input object (play and record function) and with up to 8 output objects each with its own DPT and values.	

Parameter	Settings
Attention! The end-user parameter values will only be maintained when "Overwrite end-user…" in general tab were set to "Don't overwrite".	
First scene	No **Yes**
Second scene … Tenth scene	**No** Yes
There are 10 scenes which can be individually activated here	

5.4.1 Parameter page: First scene / Tenth scene

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	
Scene number	**Scene 1** … Scene 10
Each scene can be assigned by this parameter a different input KNX scene number. Any of the 64 possible numbers can be used. The scene number to be received can be configured here. Scene 1 = value 0, Scene 2 = value 1 and so forth up to value Scene 10 = value 10.	
Possible to save scene	No **Yes**
With this selection the scene can be saved. Saving Scene 1 will requires the value 128, Scene 2 requires value 129 and so forth up to Scene 10 requires value 138 to be received in the scene input object.	
Object values are updated with	Read request to bus **Last values stored in the objects**
The values to be used when saving can be configured here, either with a read request to bus or with the last values received in the objects. Thus the user can set the desired values (e.g. using normal pushbuttons or with a visualization) of the loads and then save the new scene with a long press of the button. (according to the KNX scene standard)	
Enable / Disable object	**No** En = 1 / Dis = 0 En = 0 / Dis = 1
The function can be enabled or disabled by object when selecting this parameter. It can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa.	
Output value for event 1 … Output value for event 8	**No function** 1 bit 1 byte scaling 1 byte unsigned 1 byte signed 2 bytes unsigned 2 bytes signed 2 bytes float 4 bytes unsigned 4 bytes signed 4 bytes float
Each output can have its own DPT, even 4 byte values.	

5.5 Parameter page: Timers

Parameter	Settings
Timers	**No** Yes
The actuator can be used as a timer module with many advanced functions. It can delay any DPT or it can be used as a 1 bit very advanced staircase controller	

Parameter	Settings
Timer 1	No **Yes**
Timer 2 … Timer 10	**No** Yes
There are 10 timers which can be individually activated here.	

5.5.1 Parameter page: Timer 1 / Timer 10

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	
Timer type	Only “Reaction at OFF” Delay **Staircase** Delay and staircase

	Only ON (without delay/staircase)
The timer can be used as any of the above timer types. Only the delay can have different DPTs; the rest the of the timer trigger objects are 1 bit objects which will have different behaviours when receiving an ON or OFF respectively. This are the possible actions to be executed when the timer trigger object receives an ON (“1”): Only “Reaction at OFF”: the timer will not be executed. Delay: the function block switches ON after a time delay. Staircase: the function block immediately switches ON and stays ON for the configured staircase time and thereafter switches OFF again. Delay and staircase: the function block switches ON after a time delay and then stays ON for the configured staircase time and thereafter switches OFF again. Only ON (without delay/staircase): the function block immediately switches ON and stays ON.	

5.5.1.1 Parameter page: Timer 1 / 10 / Reaction at ON

Parameter	Settings
- Staircase time (ON duration) Base	**1 s** 5 s 10 s 1 min 5 min 10 min 1 h
- Staircase time (ON duration) Factor	**60**
Establish here the wished time for the function block to be ON The Staircase time is the period of time during which the actuator function block will be switched ON. After this time elapses, the function block switches OFF again.	
Factor changeable by object / Remaining time cyclic sending	**No** Yes

No (default option): staircase time only configurable via parameters.

Yes: this option activates an object to change staircase time factor. As you can see in the picture below, the time Base can be any of the following:

So, if you have selected, for instance, “1 s”, then the values received in this object will be in “seconds”. If you have selected “5 s” though, the values received will be in “seconds” and multiplied by 5 (base “5 s” x value received at object “10” = “50 seconds”). The same rule applies if the Base has been selected in “minutes” or “hours”.

Attention: if you send a 0 to "Timer one change staircase factor" the staircase will switch ON with a “1” and stay ON.

Additionally, to the above function, when the timer is active, this object will send the total remaining time up to 10 times with steps of 10% of the total time value until the timer finish.

In order to disable this function, the “T” flag must be deactivated.

Advanced staircase function	**No** Yes
Here the advanced functions can be activated.	

A) Parameter page: Timer 1 / 10 / Reaction at ON / Advanced staircase function

Parameter	Settings
Multiply staircase	**No** Yes
* With Yes: ***Attention! Total staircase time = staircase time x number of consecutive ON telegrams separated by less than 1 sec. from each other***	
Here you can activate the possibility to multiply the staircase time in order to extend the time during which the function block will stay ON. The total staircase ON time is calculated by taking the parameterized staircase time and multiplying it by the number of ON telegrams received. This resulting time will never exceed the parameterized maximum staircase time in the option “Maximum staircase time Base/Factor” It is important to keep in mind that the multiplication will only be done starting from the first triggering telegram (so, the Multiplying staircase function will only be executed when starting the staircase, not during execution). Therefore, these ON telegrams may not be longer than 1 second apart. Should more than 1 second elapse between two telegrams, then it will only do the multiplication of the previous pulses received. The telegrams received after this, will be ignored or interpreted as a retrigger timer function (if parameterized). Practical example: as implied by its name, the staircase time is frequently used in staircases. With the purpose of lowering the costs, instead of using a movement detector for switching ON/OFF, often push buttons are used with the staircase time as defined in the actuator. In order to save energy, the staircase time should be as short as possible, but sometimes you may wish to have the lights longer ON. In this case, this option can be very useful because it allows the end user to easily extend the staircase time by pressing several times (depending on how long the light should stay ON).	
Retrigger timer	No **Yes, excluding multiplication** Yes, including multiplication
It is possible to extend the staircase time by retriggering it (in other words, the timer starts counting again from the start). But this function will only be executed after more than 1 second has elapsed between the triggering events of the timer (if less than 1 second, see behaviour in section MULTIPLY STAIRCASE).	

No: the staircase will not be retriggered.

Yes, excluding multiplication (default option): this option will retrigger the staircase to be reset to the time (Base/Factor) as configured in the ETS application program.

For example: you have configured the staircase time in the ETS application program to be 1 minute; should the staircase time be, for instance, 1 hour as the result of a previous multiplication (Multiply staircase option), the moment you receive the retrigger telegram it will be reset to 1 minute again.

Yes, including multiplication: this option will retrigger the staircase to be reset to the current staircase time (it could be the parameterized time or the multiplied staircase time).

For example: you have configure the staircase time in the ETS application program to be 1 minute; should the staircase time be, for instance, 1 hour as the result of a previous multiplication (Multiply staircase option), the moment you receive the retrigger telegram it will be reset to 1 hour again.

Warning pulse	**No function** With own output With additional object

The warning pulse is meant to inform the end user about the fact that the staircase time is about to expire.

No function (default option): the light will go OFF without previous warning after the staircase time elapses.

With own output: the same function block will be used for this warning pulse.

The function block, according to the default parameters, the output will switch OFF 10 seconds before the end of the staircase time and it will switch ON again 2 seconds thereafter. This creates a short blinking effect as a visual warning.

It is important to be able to configure the OFF time because not all loads can switch OFF immediately (for example, lights using transformers). So, if you have selected 1 second as a warning time, it might not switch OFF at all.

With additional object: this option serves the same purpose of warning before the staircase time elapses. It is specially indicated for those places where the function block can/may not be switched ON and OFF quickly.
In these cases, the additional object can send a warning pulse to another function block just before the end of the staircase time of the main load.

Practical example: let's say this function block is used to control the flood lights of a tennis court via contactor. These lights take long to switch ON again (after they have been switched OFF), which is not energy-efficient nor practical. Therefore, to be able to generate a warning pulse, you can use an additional warning light connected to another function block, which this additional object is linked to.

1 action: ON: the additional object only sends a "1" at the configured point in time before the staircase time elapses.

2 actions : 1st OFF, 2nd ON: the additional object can execute two actions by sending:
Time before end of staircase for 1st action: a "0" at the configured point in time before the staircase time elapses.
Time before end of staircase for 2nd action: a "1" at the configured point in time before the staircase time elapses.

2 actions : 1st ON, 2nd OFF: the additional object can execute two actions by sending:

Time before end of staircase for 1st action: a "1" at the configured point in time before the staircase time elapses.
Time before end of staircase for 2nd action: a "0" at the configured point in time before the staircase time elapses.

3 actions: 1st OFF, 2nd ON, 3rd OFF (default option): the additional object can execute three actions by sending:
Time before end of staircase for 1st action: a "0" at the configured point in time before the staircase time elapses.

Time before end of staircase for 2nd action: a “1” at the configured point in time before the staircase time elapses. Time before end of staircase for 3rd action: a “0” at the configured point in time before the staircase time elapses.

5.5.1.2 Parameter page: Timer 1 / 10 / Reaction at OFF

Parameter	Settings
REACTION AT OFF	No action OFF without delay OFF with delay
Attention! Reaction at OFF cancels the running staircase	
This are the possible actions to be executed when the timer trigger object receives an OFF (“0”): No action: the timer will not be interrupted. OFF without delay (default option): the function block immediately switches OFF and the timer function is cancelled. OFF with delay: the function block switches OFF after a time delay. OFF WITH DELAY As soon as the OFF telegram is received, the Timer is cancelled.	
Object to disable timer	Yes, immediately Yes, on ending current timer **No**
The disable object will always react as follows (and cannot be otherwise configured): “1”: disable. “0”: enable. **Yes, immediately:** as soon as the Disable object receives a “1”, the timer will be cancelled and disabled. This option activates the parameter “Reaction on bus voltage recovery”. **Yes, on ending current timer:** whenever the Disable object receives a “1”, the timer will be not cancelled, but disabled. Thus, the current timer will finalize normally. This option activates the parameter “Reaction on bus voltage recovery”. **No (default option):** the disable object, including the “Reaction on bus voltage recovery” will be hidden.	

A) Parameter page: Timer 1 / 10 / Reaction at OFF / Object to disable timer

With “Object to disable timer:”
Yes, immediately
Yes, on ending current timer

Parameter	Settings
Reaction on bus voltage recovery	**Enable** Disable Last object status
On bus voltage recovery the timer can be enabled, disabled, or have the same state as before the bus failure depending on the above selection.	

5.6 Parameter page: Setpoints

Parameter	Settings
Setpoints	**No** Yes
Here the setpoints can be activated. Setpoints can be used as a two-point regulator (2 thresholds) or as a window comparator (2 thresholds + within thresholds)	

5.6.1 Parameter page: Setpoints Tab

Parameter	Settings
Practical example: Thermostat mode control by using 3 setpoints. Setpoint 1 = 22ºC > Enable value = 1 > Comfort mode Setpoint 2 = 20ºC > Enable value = 2 > Standby mode Setpoint 3 = 18ºC > Enable value = 3 > Night mode	
Setpoint 1 … Setpoint 3	No **Yes**
Thermostat controller by using the first 3 setpoints. They have been activated by default and the parameters in each setpoint have been selected individually to build a full KNX room thermostat.	
Setpoint 4 … Setpoint 10	**No** Yes
Here the individual setpoints to use as a Two-point Regulator (2 thresholds), Window comparator (2 thresholds + within thresholds) or simple thermostat can be activated.	

5.6.2 Parameter page: Setpoints 1 … 3

Parameter	Settings
Description	Setpoint 1 default parameter: **Comfort Mode Heat=22ºC, Cool=(22+2)=24ºC** Setpoint 2 default parameter: **Standby Mode Heat=20ºC, Cool=(20+6)=26ºC** Setpoint 3 default parameter: **Night Mode Heat=18ºC, Cool=(18+10)=28ºC**
This enables the integrator to add a personalized description in the text field.	
The actuator does not have a full thermostat module integrated, nevertheless by using 3 setpoints this can be achieved. In order to facilitate the understanding of how to configure the 3 setpoints they have been activated by default and the parameters in each setpoint have been selected individually to build a full KNX room thermostat. It is important to treat these 3 setpoints as “one”. Meaning that the same objects in each of the three setpoints should be linked with the same group address. E.g. to change the “HVAC mode” i.e. comfort, standby and night mode, the enable object is set to 1 byte and in each setpoint the value to enable the setpoint is different. In the example for Setpoint 1 the enable value is 1, Setpoint 2 the enable value is 2 and Setpoint 3 the enable value is 3. So if the same group address is connected to all three objects, by sending the value 1 the setpoint 1 will be enabled and the other two setpoints disabled. (all other values but the enable value disables the setpoint) To change the new current setpoint temperature one should, as previously described also connect the same group address to the three “Setpoint X setpoint value/status” objects. Only the enabled setpoint would accept the	

new setpoint change, thus unlike other room thermostats when changing the current setpoint with the same group address it always changes the value of the current selected mode. Let's have a detailed look at the default parameter example which uses the first three setpoints:

Thermostat mode control by using 3 setpoints.
1) Setpoint 1 = 22ºC > Enable value = 1 > Heat/Cool = 1 > Mode = Comfort-Heat
2) Setpoint 2 = 20ºC > Enable value = 2 > Heat/Cool = 1 > Mode = Standby-Heat
3) Setpoint 3 = 18ºC > Enable value = 3 > Heat/Cool = 1 > Mode = Night-Heat

4) Setp.1=22ºC+(2ºC Cool offset)=24ºC > Enable=1 > Heat/Cool=0 >Mode=Comfort-Cool
5) Setp.2=20ºC+(6ºC Cool offset)=26ºC > Enable=2 > Heat/Cool=0 >Mode=Standby-Cool
6) Setp.3=18ºC+(10ºC Cool offset)=28ºC > Enable=3 > Heat/Cool=0 >Mode=Night-Cool

As we can see the "Room Thermostat" can be set in 6 states. Now referring to the above states "1) - 6)" let's see what happens when sending the new setpoint value to all three setpoints at the same time.

Let's say we start off in state 1) now we send the value 21 as the new setpoint value, this will result in the following:

1) Setpoint 1 = 21ºC > Enable value = 1 > Heat/Cool = 1 > Mode = Comfort-Heat
2) Setpoint 2 = 20ºC > Enable value = 2 > Heat/Cool = 1 > Mode = Standby-Heat
3) Setpoint 3 = 18ºC > Enable value = 3 > Heat/Cool = 1 > Mode = Night-Heat

4) Setp.1=21ºC+(2ºC Cool offset)=23ºC > Enable=1 > Heat/Cool=0 >Mode=Comfort-Cool
5) Setp.2=20ºC+(6ºC Cool offset)=26ºC > Enable=2 > Heat/Cool=0 >Mode=Standby-Cool
6) Setp.3=18ºC+(10ºC Cool offset)=28ºC > Enable=3 > Heat/Cool=0 >Mode=Night-Cool

Now let's say we change to state 2) now we send the value 19 as the new setpoint value, this will result in the following:

1) Setpoint 1 = 21ºC > Enable value = 1 > Heat/Cool = 1 > Mode = Comfort-Heat
2) Setpoint 2 = 19ºC > Enable value = 2 > Heat/Cool = 1 > Mode = Standby-Heat
3) Setpoint 3 = 18ºC > Enable value = 3 > Heat/Cool = 1 > Mode = Night-Heat

4) Setp.1=21ºC+(2ºC Cool offset)=23ºC > Enable=1 > Heat/Cool=0 >Mode=Comfort-Cool
5) Setp.2=19ºC+(6ºC Cool offset)=25ºC > Enable=2 > Heat/Cool=0 >Mode=Standby-Cool
6) Setp.3=18ºC+(10ºC Cool offset)=28ºC > Enable=3 > Heat/Cool=0 >Mode=Night-Cool

Now let's say we change to state 6) now we send the value 27 as the new setpoint value, this will result in the following:

1) Setpoint 1 = 21ºC > Enable value = 1 > Heat/Cool = 1 > Mode = Comfort-Heat
2) Setpoint 2 = 19ºC > Enable value = 2 > Heat/Cool = 1 > Mode = Standby-Heat
3) Setpoint 3 = 17ºC > Enable value = 3 > Heat/Cool = 1 > Mode = Night-Heat

4) Setp.1=21ºC+(2ºC Cool offset)=23ºC > Enable=1 > Heat/Cool=0 >Mode=Comfort-Cool
5) Setp.2=19ºC+(6ºC Cool offset)=25ºC > Enable=2 > Heat/Cool=0 >Mode=Standby-Cool
6) Setp.3=17ºC+(10ºC Cool offset)=27ºC > Enable=3 > Heat/Cool=0 >Mode=Night-Cool

So as can be seen in this last step the setpoint change will always change the current setpoint status (not the parameter value) It does not matter in which KNX HVAC mode or in Heat/Cool state it is in.

This is a big advantage over most KNX room thermostats. To change the setpoint from a visualization you only need one control element to set the desired current setpoint value and it will always correspond to the current setpoint status.

Input value	**By object** Temp. sensor 1 result Temp. sensor 2 result Temp. sensor 3 result Temp. sensor 4 result Temp. sensor 5 result Temp. sensor 6 result
The reference value for the setpoint can be either one of the temperature sensors resulting values (weighted output) of the inputs or it can receive its value from the bus by selecting "By object"	

5.6.2.1 Parameter page: Setpoints 1 … 3 DPT

Parameter	Settings

Datapoint type of setpoint objects	1 byte unsigned 1 byte scaling 2 bytes unsigned **2 bytes float** 4 bytes unsigned 4 bytes float
Attention! The "… setpoint value/status" object can only be changed if the Setpoint is enabled. Initial setpoint status value if Heat/Cool modes are used: Heating = parameter value, Cooling = parameter value + "Cool offset"	
Here the DPT for both the setpoint and the hysteresis can be set. **Setpoint for most of the important DPTs (not only temperature)** This allows for instance in combination with energy meters and visualization systems to set the maximum consumption for each load and use the 4 byte values as a setpoint in order to not exceed the appointed maximum ¼ hour energy values and therefor reduce the monthly costs.	

A) Parameter page: Setpoints 1 … 3 / DPT / X bytes float

Parameter	Settings
Datapoint type of setpoint objects	… **2 bytes float** … 4 bytes float
The usual DPT for temperature values is a 2 byte float value	
Setpoint [x 0.1]	Setpoint 1 default parameter: **220** Setpoint 2 default parameter: **200** Setpoint 3 default parameter: **180**
Here the initial setpoint value can be set. It can also be changed from the bus and depending on the end-user parameters by overwritten or not when downloading with the ETS. **Higher than normal temperature setpoint value;** Using setpoints (as a thermostat) to control high setpoints temperature values (the most devices in the marked don't allow temp. setpoint higher than 45ºC. Very useful for solar panel installation control.	
Hysteresis [x 0.1]	10
Here the hysteresis value can be set.	
Type of Hysteresis (Threshold calculation)	Setpoint = Upper threshold Setpoint = Lower threshold Setpoint = Symmetric (1/2 between THs) **Heating / Cooling object**
Here the type of hysteresis for the threshold calculation can be selected. When selecting "Setpoint = Upper threshold" the Lower Threshold = Setpoint – Hysteresis (typically for heating) This is typically used for an analogue value that starts off from a lower value and when reaching the higher threshold value sends a telegram to switch the load. E.g. switch off the heating, lower the shades, etc. When selecting "Setpoint = Lower threshold" the Upper Threshold = Setpoint + Hysteresis (typically for cooling) This is typically used for an analogue value that starts off from a higher value and when reaching the lower threshold value sends a telegram to switch the load. E.g. switch off the cooling, switching on a light when getting too dark, etc.	

When selecting “Setpoint = Symmetric (1/2 between THs)” the Upper Threshold = Setpoint + ½ Hysteresis and the Lower Threshold = Setpoint - ½ Hysteresis.

When selecting “Heating / Cooling object” it switches between the first two options by sending to this object a 1 for Heating or a 0 for Cooling. In this case the “reaction exceeding…, …falling…, and …within…” cannot be selected in the parameters. It is fixed to the following:

For Heating:
Reaction exceeding/equal upper threshold = OFF
Reaction falling below/equal lower threshold = ON

For Cooling:
Reaction exceeding/equal upper threshold = ON
Reaction falling below/equal lower threshold = OFF

Send output value	**On change** Always
When selecting on change the output will only be sent the first time reaching/crossing the threshold. It will only send again when reaching/crossing the other threshold. Always on the other hand will send the output on each input event.	
Offset in setpoint for Cooling [x0.1]	Setpoint 1 default parameter: **20** Setpoint 2 default parameter: **60** Setpoint 3 default parameter: **100**
Here the offset of the setpoint temperature when changing to the cool mode can be selected. Example: Assuming the setpoint is 22ºC When the value in this parameter is 20 (2K), then the setpoint for cooling will be 22 + 2 = 24ºC	
Enable / disable function	**No** Yes
The setpoint can be enabled or disabled by object when selecting this parameter.	
Attention! The end-user parameter values will only be maintained when “Overwrite end-user…” in general tab were set to “Don’t overwrite”.	

A.1) Parameter page: Setpoints 1 … 3 / DPT/ X bytes float / Enable / Disable function

Parameter	Settings
Enable / disable object	1 bit **1 byte unsigned**
The setpoint can be enabled with a 1 bit on/off telegram or with a 1 byte unsigned telegram. The latter can be used for instance to set the HVAC mode.	

Enable / Disable	Setpoint 1 default parameter: **1** Setpoint 2 default parameter: **2** Setpoint 3 default parameter: **3**
When selecting 1 bit, it can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa. When selecting 1 byte to enable the setpoint, the enable value can be set in the parameters. When sending this enable value to the object the setpoint will be enabled, any other value disables the setpoint. When using it for the HVAC mode use one of the following enable values: Comfort mode = 1 Standby mode = 2 Night/saving mode = 3 Frost/Heat protection = 4	
- Reaction on bus voltage recovery	**Enable** Disable Last object status
Whether the setpoint will be active or not on bus voltage recovery can be configured here. On bus voltage recovery the setpoint can be enabled, disabled, or have the same state as before the bus failure depending on the above selection. **Enable:** the setpoint will be enabled. **Disable:** the setpoint will be disabled. **Last object status:** the status of the Enable object will be saved in the actuator's non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will set the object as it was before the bus failure.	
Reaction of output and setpoint at enabling	Nothing Set calculated output Send setpoint **Both**
The reaction of output and setpoint at enabling can be selected to send the Send setpoint, Set calculated output or both the former. This is especially useful to control Air Condition systems as additional heating and/or cooling. Most KNX thermostats don't send the setpoint values with each change (heat/cool, Comfort/Standby/…) to the bus. In order to control a Split unit as an additional cooling via a gateway it is essential to send the new setpoint on each and every change.	
Reaction of output and setpoint at disabling	**Block and send nothing** Block and set output to 0 and send
The reaction of output and setpoint at disabling can be selected to block and send nothing or to block and set output to 0 and send the setpoint value. This is also useful for the above example.	

5.6.3 Parameter page: Setpoints 4 … 10

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	
Input value	**By object** Temp. sensor 1 result

	Temp. sensor 2 result Temp. sensor 3 result Temp. sensor 4 result Temp. sensor 5 result Temp. sensor 6 result
The reference value for the setpoint can be either one of the temperature sensors resulting values (weighted output) of the inputs or it can receive its value from the bus by selecting “By object”	

5.6.3.1 Parameter page: Setpoints 4 … 10 DPT

Parameter	Settings
Datapoint type of setpoint objects	1 byte unsigned 1 byte scaling 2 bytes unsigned **2 bytes float** 4 bytes unsigned 4 bytes float
Attention! The “… setpoint value/status” object can only be changed if the Setpoint is enabled. Initial setpoint status value if Heat/Cool modes are used: Heating = parameter value, Cooling = parameter value + “Cool offset”	
Here the DPT for both the setpoint and the hysteresis can be set. **Setpoint for most of the important DPTs (not only temperature)** This allows for instance in combination with energy meters and visualization systems to set the maximum consumption for each load and use the 4 byte values as a setpoint in order to not exceed the appointed maximum ¼ hour energy values and therefor reduce the monthly costs.	

A) Parameter page: Setpoints 4 … 10 / DPT / X bytes float

Parameter	Settings
Datapoint type of setpoint objects	… **2 bytes float** … 4 bytes float
Setpoint [x 0.1]	220
Here the initial setpoint value can be set. It can also be changed from the bus and depending on the end-user parameters be overwritten or not when downloading with the ETS. **Higher than normal temperature setpoint value;** Using setpoints (as a thermostat) to control high setpoints temperature values (the most devices in the marked don’t allow temp. setpoint higher than 45ºC. Very useful for solar panel installation control.	
Hysteresis [x 0.1]	10
Here the hysteresis value can be set.	
Type of Hysteresis (Threshold calculation)	**Setpoint = Upper threshold** Setpoint = Lower threshold Setpoint = Symmetric (1/2 between THs) Heating / Cooling object

Here the type of hysteresis for the threshold calculation can be selected.

When selecting “Setpoint = Upper threshold” the Lower Threshold = Setpoint – Hysteresis (typically for heating)

This is typically used for an analogue value that starts off from a lower value and when reaching the higher threshold value sends a telegram to switch the load. E.g. switch off the heating, lower the shades, etc.

When selecting “Setpoint = Lower threshold” the Upper Threshold = Setpoint + Hysteresis (typically for cooling)

This is typically used for an analogue value that starts off from a higher value and when reaching the lower threshold value sends a telegram to switch the load. E.g. switch off the cooling, switching on a light when getting too dark, etc.

When selecting “Setpoint = Symmetric (1/2 between THs)” the Upper Threshold = Setpoint + ½ Hysteresis and the Lower Threshold = Setpoint - ½ Hysteresis.

When selecting “Heating / Cooling object” it switches between the first two options by sending to this object a 1 for Heating or a 0 for Cooling. In this case the “reaction exceeding…, …falling…, and …within…” cannot be selected in the parameters. It is fixed to the following:
For Heating:
Reaction exceeding/equal upper threshold = OFF
Reaction falling below/equal lower threshold = ON
For Cooling:
Reaction exceeding/equal upper threshold = ON
Reaction falling below/equal lower threshold = OFF

Reaction exceeding/equal upper threshold	No reaction On **Off** On, first time exceeding Off, first time exceeding
Here the reaction exceeding/equal upper threshold can be set.	
Reaction falling below/equal lower threshold	No reaction **On** Off On, first time falling below Off, first time falling below
Here the reaction falling below/equal lower threshold can be set.	
Reaction within threshold	**No reaction** On Off On, first time entering Off, first time entering
Here the reaction within threshold can be set	
Enable / disable function	**No** Yes
The setpoint can be enabled or disabled by object when selecting this parameter.	
Attention! The end-user parameter values will only be maintained when “Overwrite end-user…” in general tab were set to “Don’t overwrite”.	

A.1) Parameter page: Setpoints 4 … 10 / DPT/ X bytes float / Enable / Disable function

Parameter	Settings
Enable / disable object	**1 bit** 1 byte unsigned
The setpoint can be enabled with a 1 bit on/off telegram or with a 1 byte unsigned telegram. The latter can be used for instance to set the HVAC mode.	
Enable / Disable	**En =1 / Dis = 0** En =0 / Dis = 1
When selecting 1 bit, it can be configured to enable with an ON telegram and to disable with an OFF telegram or vice versa. When selecting 1 byte to enable the setpoint, the enable value can be set in the parameters. When sending this enable value to the object the setpoint will be enabled, any other value disables the setpoint. When using it for the HVAC mode use one of the following enable values: Comfort mode = 1 Standby mode = 2 Night/saving mode = 3 Frost/Heat protection = 4	
- Reaction on bus voltage recovery	**Enable** Disable Last object status
Whether the setpoint will be active or not on bus voltage recovery can be configured here. On bus voltage recovery the setpoint can be enabled, disabled, or have the same state as before the bus failure depending on the above selection. **Enable:** the setpoint will be enabled. **Disable:** the setpoint will be disabled. **Last object status:** the status of the Enable object will be saved in the actuator's non-volatile memory; therefore, when the actuator initializes, if this option has been chosen, it will set the object as it was before the bus failure.	
Reaction of output and setpoint at enabling	**Nothing** Set calculated output Send setpoint Both
The reaction of output and setpoint at enabling can be selected to send the Send setpoint, Set calculated output or both the former. This is especially useful to control Air Condition systems as additional heating and/or cooling. Most KNX thermostats don't send the setpoint values with each change (heat/cool, Comfort/Standby/…) to the bus. In order to control a Split unit as an additional cooling via a gateway it is essential to send the new setpoint on each and every change.	
Reaction of output and setpoint at disabling	**Block and send nothing** Block and set output to 0 and send
The reaction of output and setpoint at disabling can be selected to block and send nothing or to block and set output to 0 and send the setpoint value. This is also useful for the above example.	

5.7 Parameter page: Internal variables

Parameter	Settings
Internal variables	**No** Yes
This can be used to make internal links like the links done by using group addresses but with the main difference that they are not sent to the bus. Only output objects can be linked to input objects. Care should be taken to link only objects with the same DPT, this must be checked by the integrator, it is not checked by the application program. Should they have different sizes it will not work.	

Parameter	Settings
Internal variables 1…10	No **Yes**
Internal variables 11…20 Internal variables 21…30 Internal variables 31…40 Internal variables 41…50	**No** Yes
Attention! *It is recommended to only use variables for internal links. If group addresses are also linked, execution will take longer.*	
A total of 50 internal links can be done	

5.7.1 Parameter page: Variables 1…10

Parameter	Settings
Description	
This enables the integrator to add a personalized description in the text field.	

Parameter	Settings
Variable 1	No **Yes**
Variable 2 … Variable 10	**No** Yes
There are a total of 10 variable per page	

5.7.2 Parameter page: Variables 1…10 / Input object

Parameter	Settings

Input object to send variable	General **Function blocks** Alarms Logic Scene controller Timers Setpoints
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	

Parameter	Settings
Input object to send variable	**General**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Object name	**Central cyclic telegram for monitoring** Telegram at bus recovery
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	

Parameter	Settings
Input object to send variable	**Function blocks**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select function blocks	**A1** A2 B1 B2 C1 C2
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Function block input** Function block input toggle / inverted RunHour counter threshold RunHour counter reset Switching counter threshold Switching counter reset Scene number Scene disable Timer 1 trigger Timer 1 change staircase factor Timer 1 disable Timer 2 trigger Timer 2 change staircase factor Timer 2 disable Disable function block
In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Input object to send variable	**Alarms**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select alarm	**Alarm 1** … Alarm 8
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Alarm** Alarm setpoint Alarm hysteresis Alarm disable
In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Input object to send variable	**Logics**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select logic	**Logic 1** … Logic 20
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Logic disable** Logic input 1 Logic input 2 / Enable Gate Logic input 3 Logic input 4
In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Input object to send variable	**Scenes**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select KNX scene	**Scene 1** … Scene 10
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Scene input** Scene disable

In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.

Parameter	Settings
Input object to send variable	**Timers**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select timer	**Timer 1** … Timer 10
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Timer trigger** Timer change staircase factor Timer disable
In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Input object to send variable	**Setpoints**
In order to find and select the input object to be linked with the output object one has different filters. This is the main filter where all main functions of the actuator are listed. (Except for the inputs – they cannot be linked with internal variables)	
Select setpoint	**Setpoint 1** … Setpoint 10
In order to find and select the input object to be linked with the output object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Setpoint Heat / Cool** Setpoint disable Setpoint value/status Setpoint input ext. sensor value
In order to find and select the input object to be linked with the output object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

5.7.1 Parameter page: Variables 1…10 / Output object

Parameter	Settings
Output object to send variable	General **Function block** Logic Scene controller Timers Setpoints
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with internal variables)	

Parameter	Settings
Output object to send variable	General
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with internal variables)	
Object name	**Central cyclic telegram for monitoring** Telegram at bus recovery
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	

Parameter	Settings
Output object to send variable	**Function block**
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with	

internal variables)

Select function block	**A1** A2 B1 B2 C1 C2
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Switching status** RunHour counter RunHour counter alarm RunHour counter value at reset Switching counter Switching counter alarm Switching counter value at reset Timer 1 warning pulse Timer 2 warning pulse
In order to find and select the output object to be linked with the input object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Output object to send variable	**Logics**
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with internal variables)	
Select logic	**Logic 1** … Logic 20
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Logic output**
In order to find and select the output object to be linked with the input object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Output object to send variable	**Scene controller**
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with internal variables)	
Select scene	**Scene 1** … Scene 10
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Scene controller event 1** … Scene event 8

In order to find and select the output object to be linked with the input object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.

Parameter	Settings
Output object to send variable	Timers
In order to find and select the output object to be linked with the input object one has different filters. This is the main filter where all main functions of the actuator are listed. (except for the inputs – they cannot be linked with internal variables)	
Select timer	**Timer 1** … Timer 10
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Timer warning pulse** Timer output
In order to find and select the output object to be linked with the input object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

Parameter	Settings
Output object to send variable	Setpoints
Select setpoint	**Setpoint 1** … Setpoint 10
In order to find and select the output object to be linked with the input object one has different filters. This is the first sub-filter where all the sub functions of the previously selected main function of the actuator are listed.	
Object name	**Setpoint output regulator**
In order to find and select the output object to be linked with the input object one has different filters. This is the second sub-filter where all the secondary sub functions of the previously selected sub-function of the actuator are listed.	

5.8 Parameter page: Overwrite end-user parameter values at download

Parameter	Settings
Overwrite end-user parameter values at download	No **Yes** Custom
It is very important for the end user to be able to change (via dedicated objects linked, for instance, to a visualization) certain settings of his/her KNX installation. This actuator allows for these changes to be maintained even when downloading the application program with the ETS again. If no end-user parameters should be downloaded the “No” option should be selected. But it is also possible by selecting “**Custom**” to individually decide whether or not the end-user parameters should be downloaded.	

5.8.1 Parameter page: Enduser Parameter

5.8.1.1 Parameter page: Enduser Parameter / Advanced Functions

A) Parameter page: ADVANCED FUNCTIONS / Alarms

Parameter	Settings
Alarms	**Overwrite complete module** Overwrite individually Don’t overwrite
If none of the Alarm end-user parameters should be downloaded the “Don’t overwrite” option should be selected. But it is also possible by selecting “Overwrite individually” to individually decide whether or not the end-user parameters of any one of the 8 Alarms should be downloaded.	

A.1) Parameter page: ADVANCED FUNCTIONS / Alarms / Overwrite individually

Parameter	Settings
Alarms	**Overwrite individually**
- Alarm 1 … - Alarm 8	**Overwrite** Don’t overwrite
Select here whether to overwrite or not	

B) Parameter page: ADVANCED FUNCTIONS / Scenes

Parameter	Settings
Scenes	**Overwrite complete module** Overwrite individually Don't overwrite
If none of the Scene end-user parameters should be downloaded the "Don't overwrite" option should be selected. But it is also possible by selecting "Overwrite individually" to individually decide whether or not the end-user parameters of any one of the 10 scenes should be downloaded.	

B.1) Parameter page: ADVANCED FUNCTIONS / Scenes / Overwrite individually

Parameter	Settings
Scenes	**Overwrite individually**
- First scene … - Tenth scene	**Overwrite** Don't overwrite
Select here whether to overwrite or not	

C) Parameter page: ADVANCED FUNCTIONS / Timers

Parameter	Settings
Timers	**Overwrite complete module** Overwrite individually Don't overwrite
If none of the Timers end-user parameters should be downloaded the "Don't overwrite" option should be selected. But it is also possible by selecting "Overwrite individually" to individually decide whether or not the end-user parameters of any one of the 10 Timers should be downloaded.	

C.1) Parameter page: ADVANCED FUNCTIONS / Scene controller / Overwrite individually

Parameter	Settings
Timers	**Overwrite individually**
- Timer 1 … - Timer 10	**Overwrite** Don't overwrite
Select here whether to overwrite or not	

D) Parameter page: ADVANCED FUNCTIONS / Setpoints

Parameter	Settings

Setpoints	**Overwrite complete module** Overwrite individually Don't overwrite
If none of the Setpoints end-user parameters should be downloaded the "Don't overwrite" option should be selected. But it is also possible by selecting "Overwrite individually" to individually decide whether or not the end-user parameters of any one of the 10 Setpoints should be downloaded.	

D.1) Parameter page: ADVANCED FUNCTIONS / Setpoints / Overwrite individually

Parameter	Settings
Setpoints	**Overwrite individually**
- Setpoint 1 … - Setpoint 10	**Overwrite** Don't overwrite
Select here whether to overwrite or not	

5.8.1.2 Parameter page: ENDUSER PARAMETERS / FUNCTION BLOCKS

Parameter	Settings
FUNCTION BLOCK	**Overwrite all function blocks** Overwrite individually Don't overwrite
If the function blocks end-user parameters should be downloaded the "Don't overwrite" option should be selected. But it is also possible by selecting "Overwrite individually" to individually decide whether or not the end-user parameters of any one of the function block parameters should be downloaded.	

A) Parameter page: ENDUSER PARAMETERS / FUNCTION BLOCKS / Function block A1… D2

Parameter	Settings
FUNCTION BLOCK	**Overwrite individually**
- Scenes	**Overwrite** Don't overwrite
Select here whether to overwrite or not	
- Counters	**Overwrite** Don't overwrite
Select here whether to overwrite or not	

B) Parameter page: ENDUSER PARAMETERS / FUNCTION BLOCKS / Function block A1… D2 / only binary

Parameter	Settings
FUNCTION BLOCK	**Overwrite individually**

- Scenes	**Overwrite** Don’t overwrite
Select here whether to overwrite or not	
- Counters	**Overwrite** Don’t overwrite
Select here whether to overwrite or not	

5.9 Parameter page: Central sending object for monitoring device

Parameter	Settings
Central sending object for monitoring device	**No** Yes
This activates a central cyclic sending object which can be used to monitor if the device is still sending this telegram. This way a KNX line and or the actuator can be supervised if they are still reachable.	

Parameter	Settings
- Sending period (0=only answer) min.	**0**
The cyclic sending rate can be introduced here, should the object be polled it is not necessary to send it cyclically and therefore it can be set to zero. Then this object will only answer to read requests.	

5.10 Parameter page: Behaviour at bus recovery

Parameter	Settings
Behaviour at bus recovery	**No** Yes
The behaviour at bus voltage failure and recovery can be established in most parts (function blocks, inputs, advanced functions) in the application program of the actuator, but the sending delays and frequencies can be adjusted here.	

Parameter	Settings
- Send telegram for external use	**No** Yes
It is very usual to have to do different actions when the KNX devices are powered up, like a scene to establish some default parameters (establish temperature setpoint values, trigger a scene, reset a variable, etc...). By activating this function the actuator will send a telegram with a fixed value to the bus after bus recovery. The DPT can also be selected to be: 1 bit, 1 byte unsigned, 1 byte scaling and 2 byte float.	
- Delay for sending all status telegrams	Immediately 1 s **5 s** 10 s 20 s 30 s 1 min 3 min 5 min 10 min
The behaviour at bus voltage failure and recovery can be established in most parts (function blocks, inputs, advanced functions) in the application program of the actuator, which could cause generating status telegrams after recovery of the bus voltage, but some devices might take longer to start-up (like touch displays, visualization servers, etc.). In these cases the delay for sending the status telegrams can be set here.	
Delay for all initial read request and execute on init commands	Immediately 1 s 5 s **10 s** 20 s 30 s 1 min 3 min 5 min 10 min
The delay for all initial read request and execute on initialization commands can be set here.	
- Delay between read request / status telegrams	Immediately **500 ms** 1 s 2 s
Should the behaviour on bus voltage return be configured in many places in the actuator, this could cause multiple telegrams to the bus be sent at the same time. For this not to happen one can select here the delay between telegrams sent to the bus after bus recovery.	

6 Firmware version and update

If there is a new firmware available, it can be updated via a micro SD card in only a couple of seconds.

Procedure:
1) Remove the bus connector of the device leaving it without bus voltage.
2) Copy the xxxxx.bin (e.g. for the InBlock i8 HV device the file would be: P5_i8.bin) file to the micro SD card and put it into the micro SD card slot of the device.
3) Press the ETS physical address programming button next to the bus connector of the device
4) Without releasing the button plug in the bus connection while maintaining to hold the button until the programming LED starts to flash and then release it (before it stops to flash)
5) Finished! Now the ETS application program can be download by using the normal procedure using the ETS.

Attention! Never insert the micro SD Card when the device is connected to the KNX bus voltage! This could cause the device to reset without storing the variables previously to the Flash memory. Thus all these variables (e.g. counter values, scene values …) will be lost.

7 Reset to conditions at delivery

To reset the device to its original settings, repeat the same procedure as above using the last valid firmware.

This leads to a factory reset. All device settings return to their status at delivery and the device has the physical address 15.15.255.

8 Annex

8.1 Annex 1: Manual Control

The **inputs** of the InBlock have 1 push button and 1 status LED for each input on the below LED row
These buttons can be activated to control each and every input individually if you select “yes” in the relevant parameter options in Binary Input.
The LEDs represent: The below row inputs 1&4, 2&5, 3&6 actual input status

PARAMETER MODE

MANUAL CONTROL – PARAMETER MODE The Parameter Mode allows you to control all the inputs in the device as configured in the ETS. The Action simulates a closed contact in order to send a telegram via input object of the selected one.
BINARY INPUT
Press action on 1 & 4, 2 & 5, 3 & 6: Sends Toggle ON/OFF command 0/1 to the “associated object” of the input (simulates the close/open action on the binary contact)  LED = ON (indicates input status -> Input contact closed)  LED = OFF (indicates channel status -> Input contact open) “Man” push button in the right side for selection inputs status range between input 1..3 (LED = OFF) and inputs 4..6 (LED = Blinking)

8.2 Annex 2: Flowcharts