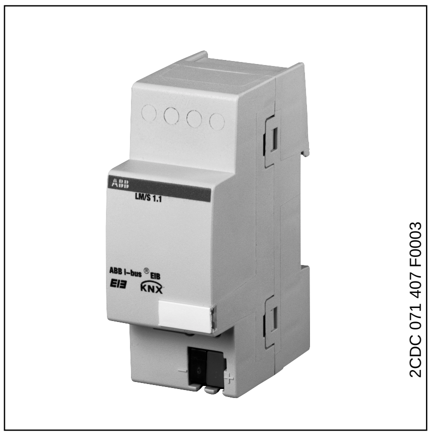
# Logic module, MDRC
# LM/S 1.1, GH Q631 0080 R0111



The logic module is a DIN rail mounted device for insertion in the distribution board. The connection to the EIB is carried out via a bus connecting terminal.

The logic module is used in more complex EIB installations in order to create special logic operations between sensors and actuators.

At least 3 logic or time functions can always be set in each logic module both simultaneously and independently of each other.

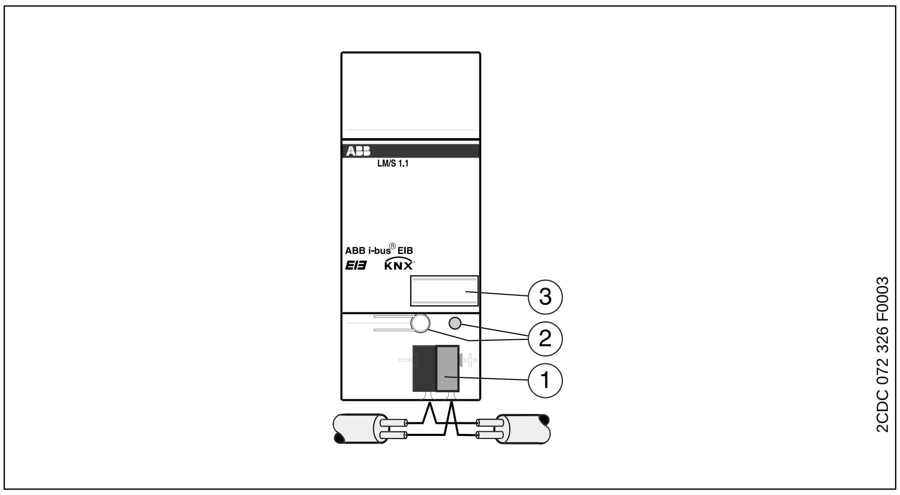
The following functions can be implemented with the logic module:
– Logic gates
– Gate/filter
– Time delay
– Multiplier
– Min./Max. value indication
– Temperature comparator
– Toggle value
– Threshold value
– Format converter
– Scenes
– Counter
– Staircase light

**Technical data**

| | | |
|---|---|---|
| **Power supply** | – EIB | 24 VDC, via the bus line |
| **Operating and display elements** | – LED (red) and push button | for assigning the physical address |
| **Type of protection** | – IP 20, EN 60 529 | |
| **Protection class** | – II | |
| **Ambient temperature range** | – Operation | - 5 °C … 45 °C |
| **Connection** | – EIB | Bus connecting terminal included with supply |
| **Installation** | – on 35 mm mounting rail, DIN EN 60 715 | |
| **Dimensions** | – 90 x 36 x 64 mm (H x W x D) | |
| **Weight** | – 0.1 kg | |
| **Certification** | – EIB-certified | |
| **CE norm** | – in accordance with the EMC guideline and the low voltage guideline | |

| Application programs | Number of communication objects | Max. number of group addresses | Max. number of associations |
|---|---|---|---|
| Logic Threshold Scene /1 | 48 | 250 | 250 |
| | | | |

**Circuit diagram**

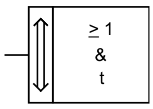


**1** Bus connection
**2** Programming LED, button
**3** Label

**Note**

To commission the logic module, you need a PC with the Engineering Tool Software ETS (from ETS2 V1.2a onwards) and an interface to the EIB (e.g. via an RS232 interface).

**Logic Threshold Scene /1**



**Selection in ETS2**

– ABB
 └Controller
  └Controller

Each EIB installation has its own requirements. The logic module makes 12 different functions available in one device for special logic operations between sensors and actuators:
– Logic gates
– Gate/filter
– Time delay
– Multiplier
– Min./Max. value indication
– Temperature comparator
– Toggle value
– Threshold value
– Format converter
– Scenes
– Counter
– Staircase light

Each logic module can always carry out three of the functions listed above simultaneously. The functionality of the three functions A, B and C is defined on the “General” parameter page.

The functionality of function A for example is defined via the setting “Function A”. Different parameters and communication objects are displayed depending on the setting.

It is possible to assign the different functions with any or the same functionality. The logic module can therefore be adapted to a wide variety of applications.

The individual functions are described in the following section. All the descriptions refer to function A as well as functions B and C.

Format converter

The following bit sizes can be changed with the “Format converter” function:

| | | |
|---|---|---|
| – 2x1-Bit | --> | 1x2-Bit |
| – 8x1-Bit | --> | 1-Byte |
| – 1x1-Byte | --> | 2-Byte |
| – 2x1-Byte | --> | 2-Byte |
| – 2x2-Byte | --> | 4-Byte |
| – 1x2-Bit | --> | 2x1-Bit |
| – 1-Byte | --> | 8x1-Bit |
| – 2-Byte | --> | 2x1-Byte |
| – 4-Byte | --> | 2x2-Byte |

With the parameter “Converter function”, it is defined which data types should be transferred. The ETS program displays different communication objects depending on the setting.

In general, the first bit/byte (0) always has the lowest value while the last bit/ byte (e.g. 7) always has the highest value. The conversion should be explained in more detail using the following example.

**Note:**
Telegrams are broken down or compiled on the format level. It is therefore not possible to assign a specific EIS type to the input and output objects.

An association between the object and an EIS type first originates from the context of the ETS project, in which this function is used.

**Example: 2x1-Bit --> 1x2-Bit**
Input:

| | |
|---|---|
| $2^1$ | $2^0$ |
| Bit1 | Bit0 |
| 1 | 1 |

Output:
2 bit value: $3_{dec}$

**Example: 1-Byte --> 8x1-Bit**
Input:
1 byte value: $169_{dec}$

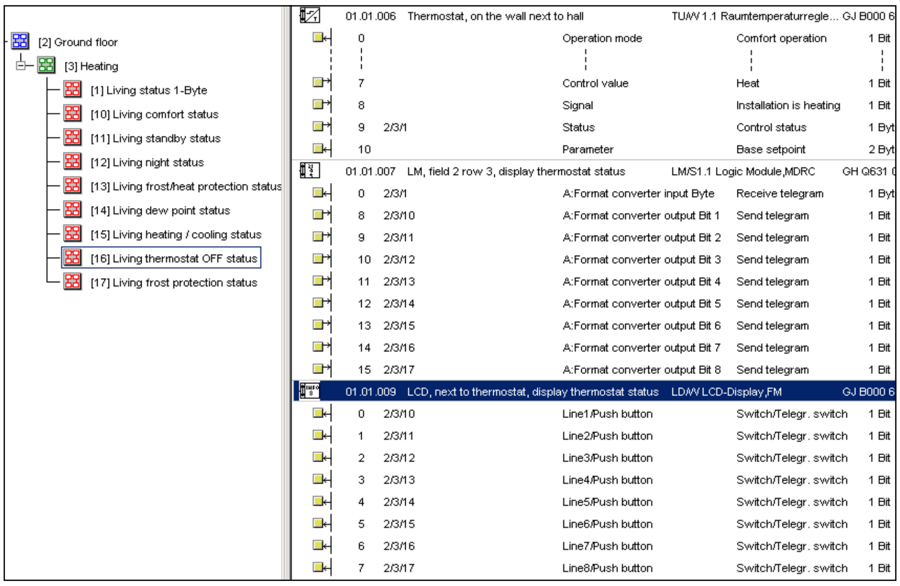
Output:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |

The conversion of 8x 1 bit into 1 byte is carried out in the same way.

**Example of a format converter**
1 byte --> 8x1 bit

The 1 byte status value is issued by the room thermostat and converted into 8x1 bit variables in the logic module. They can be shown on the LC display.



```
[2] Ground floor
  [3] Heating
    [1] Living status 1-Byte
    [10] Living comfort status
    [11] Living standby status
    [12] Living night status
    [13] Living frost/heat protection status
    [14] Living dew point status
    [15] Living heating / cooling status
    [16] Living thermostat OFF status
    [17] Living frost protection status
```

```
01.01.006  Thermostat, on the wall next to hall             TU/W 1.1 Raumtemperaturregle...  GJ B000 6
    0              Operation mode                    Comfort operation          1 Bit
    7              Control value                     Heat                       1 Bit
    8              Signal                            Installation is heating    1 Bit
    9   2/3/1      Status                            Control status             1 Byt
    10             Parameter                         Base setpoint              2 Byt
01.01.007  LM, field 2 row 3, display thermostat status     LM/S1.1 Logic Module,MDRC       GH Q631 0
    0   2/3/1      A:Format converter input Byte     Receive telegram           1 Byt
    8   2/3/10     A:Format converter output Bit 1   Send telegram              1 Bit
    9   2/3/11     A:Format converter output Bit 2   Send telegram              1 Bit
    10  2/3/12     A:Format converter output Bit 3   Send telegram              1 Bit
    11  2/3/13     A:Format converter output Bit 4   Send telegram              1 Bit
    12  2/3/14     A:Format converter output Bit 5   Send telegram              1 Bit
    13  2/3/15     A:Format converter output Bit 6   Send telegram              1 Bit
    14  2/3/16     A:Format converter output Bit 7   Send telegram              1 Bit
    15  2/3/17     A:Format converter output Bit 8   Send telegram              1 Bit
01.01.009  LCD, next to thermostat, display thermostat status  LD/W LCD-Display,FM          GJ B000 6
    0   2/3/10     Line1/Push button                 Switch/Telegr. switch      1 Bit
    1   2/3/11     Line2/Push button                 Switch/Telegr. switch      1 Bit
    2   2/3/12     Line3/Push button                 Switch/Telegr. switch      1 Bit
    3   2/3/13     Line4/Push button                 Switch/Telegr. switch      1 Bit
    4   2/3/14     Line5/Push button                 Switch/Telegr. switch      1 Bit
    5   2/3/15     Line6/Push button                 Switch/Telegr. switch      1 Bit
    6   2/3/16     Line7/Push button                 Switch/Telegr. switch      1 Bit
    7   2/3/17     Line8/Push button                 Switch/Telegr. switch      1 Bit
```

The conversion of 1x1 byte into 1x2 byte represents a special case. The input byte in (EIS6) format is transferred into a 2 byte value (EIS10). It is possible for example with this conversion to combine the 1 byte output of the EIB current module with the 2 byte input value of the analogue actuator.

When converting from 2x1 byte into a 2 byte value or from a 2 byte value into 2x1 byte values, the 1 byte values form a low byte and a high byte. The low byte is the lowest value byte and creates bit numbers 0 to 7 ($2^0$ to $2^7$) while the high byte creates bit numbers 8 to 15 ($2^8$ to $2^{15}$).

The above diagram shows an application example for the format converter. The 1 byte controller status which is sent by a room thermostat should be indicated on an LC display. First of all, the parameter “Converter function” of function A is set to “1-Byte --> 8x1-Bit”. The 1 byte communication object of the room thermostat is then linked to the 1 byte input object of the format converter. The eight 1 bit outputs are linked to the eight 1 bit inputs of the LC display.

**Bus voltage recovery**

Since not all telegrams are generally sent immediately to the inputs after a bus voltage recovery, it is possible in this case to read out the input value or input values via the bus in order to put the input in a defined state.

If the parameter “Read input values over EIB after bus voltage recovery” is set to “Yes” for this purpose, the logic module waits 5 seconds after starting up and then sends corresponding read telegrams to the linked group addresses.

**Send output values**

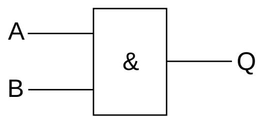
Under “Send output values”, it can be selected when an output telegram should be sent. If it is set to “on object value change”, an output telegram is only sent when the value of the object has changed e.g. from “0” to “1”. If the output should also be sent each time a telegram is received at one of the inputs, the option “on reception of new telegram” must be set. In this case, an output telegram is also sent if the value of the output has not been changed.

**Logic gates with truth tables**
The logic gates listed on the right can be set. The gates and the tables describe the input and output states for only 2 inputs. If there are several inputs, the tables must be extended accordingly.

[1] If there is an uneven number of inputs which are set to ON (receive ON telegram), the output is also always ON (sends an ON telegram).

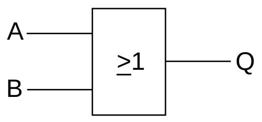
[2] If there is an uneven number of inputs which are set to OFF (receive OFF telegram), the output is also always ON (sends an ON telegram).

AND



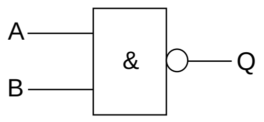
| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

OR



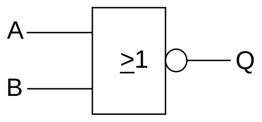
| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

NAND (NOT-AND)



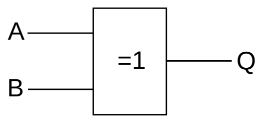
| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

NOR (NOT-OR)



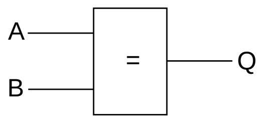
| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 0 |

XOR (Exclusive-OR)[1]



| A | B | Q |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

XNOR (Equivalence)[2]



| A | B | Q |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

## Logic gates

If the “Logic” function is selected in the “General” window, the ETS parameters and communication objects indicate which logic functions can be implemented. These logic functionalities are required e.g. to set up more complex logic operations which can no longer be implemented with the standard logic operations of switch or shutter actuators.

The diagram above shows all the possible logic operations with the associated truth tables. Via the parameter “Logical operation”, it is possible to choose between the following functions:
– AND
– OR
– Exclusive OR
– NAND
– NOR
– Equivalence

**Inputs**

A maximum of 8 inputs can be isolated for each function. This is carried out via the setting “Input ...”. Inputs 1 to 4 are shown as “normal” by default. Alternatively, the inputs can also be switched to “inverted” or “inactive” (“not used”). Inputs 5 to 8 are inactive by default (“not used”). Inputs 5 to 8 can likewise be isolated (“normal”) or inverted via the setting “Input ...”.

ETS displays a unique 1 bit communication object for each active input. A group address is linked to this object. After each receipt of a telegram at one of a maximum of 8 inputs, the output signal is regenerated.

**Output**

The output of the logic operation lies on the 1 bit communication object “...: Logic output - Send telegram”. This signal can be inverted via the setting “Output”. It is defined with the parameter “Send output values” when the output should send out a telegram. This can be “on reception of new telegram” (the output is sent after receipt of an input telegram, regardless of whether the output value has changed or not) or only “on object value change” (after receipt of an input telegram that calls up a change in the output signal).

It is possible to also set a send delay in the two settings above for the send behaviour of the output. This means that the output signal is only sent once this delay has elapsed. The delay is composed of a base and a factor.
*Send delay = Base * Factor*

If the output value should change again within the send delay, the “old” output value is no longer sent. The logic module adopts the new output value and restarts the send delay.

Alternatively, the output value can be sent cyclically. To do so, the parameter “Send output values” must be set to “cyclic”. The cyclic interval with which the output value is sent is composed of a base value and a factor.
*Cyclic interval = Base * Factor*

**Bus voltage recovery**

Since not all telegrams are generally sent immediately to the inputs after a bus voltage recovery, it is possible in this case to use either “use value before bus voltage fail” or “read values over EIB” in order to put the input in a defined state. It is also possible to select “use 0 as preset” or “use 1 as preset” to define the input.

## Min./Max. value indication

If the functionality “Min./Max. value indication” is selected, the ETS program displays 4 input communication objects and 1 output communication object. The parameter “Data type of the inputs” defines the size of the communication objects. They can be the following types:

| | |
|---|---|
| – 1-Byte Value in % | EIS6 |
| – 1-Byte Orientation in degrees | EIS6 |
| – 1-Byte Counter (0…255) | EIS14 |
| – 1-Byte Counter (-128…+127) | EIS14 |
| – 2-Byte Counter (0…65.535) | EIS10 |
| – 2-Byte Floating point value | EIS5 |
| – 2-Byte Counter (-32.768…+32.767) | EIS10 |
| – 4-Byte Floating point value | EIS9 |
| – 4-Byte Counter (-10.000.000…+10.000.000) | EIS11 |

Each of the 4 inputs can be enabled (“used”) or disabled (“not used”) via a separate parameter “Input ...”.

Depending on the setting of the “Output” parameter, the output either adopts the highest value, the lowest value or the average.

With the parameter “Send output values”, it is defined when the output should send a telegram. This can be “on reception of new telegram” (the output is sent after receipt of an input telegram, regardless of whether the output value has changed or not) or only “on object value change” (after receipt of an input telegram that calls up a change in the output signal).

Alternatively, the output value can be sent cyclically. To do so, the parameter “Send output values” must be set to “cyclic”. The cyclic interval with which the output value is sent is composed of a base value and a factor.
*Cyclic interval = Base * Factor*

## Threshold values

The function “Threshold values” is used to detect an overrange or underflow in a set threshold value. For example, the temperature of a cold-storage depot can be monitored for a maximum value and a signal can be sent when the temperature is exceeded.

If the “Threshold values” function is selected, two independent threshold values are available. Both have their own set of parameters and communication objects. The size of the input communication object is defined via the setting “Data type of the input”. The following types are available for selection:

| | |
|---|---|
| – 1-Byte Value in % | EIS6 |
| – 1-Byte Orientation in degrees | EIS6 |
| – 1-Byte Counter (0…255) | EIS14 |
| – 1-Byte Counter (-128…+127) | EIS14 |
| – 2-Byte Counter (0…65.535) | EIS10 |
| – 2-Byte Floating point value | EIS5 |
| – 2-Byte Counter (-32.768…+32.767) | EIS10 |
| – 4-Byte Floating point value | EIS9 |
| – 4-Byte Zähler (-10.000.000…+10.000.000) | EIS11 |

Each threshold value output has its own 1 bit communication object available. An ON or OFF telegram is sent via this object when the value exceeds or falls below the threshold value.

The threshold value which should trigger a telegram is defined via the parameter “Threshold value”. The absolute value is entered here.

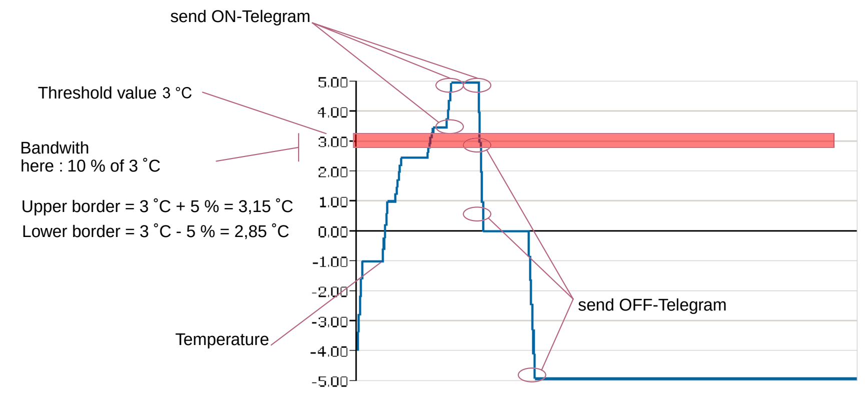
The parameter “Bandwidth of the hysteresis in % of the threshold value [0...50%]” is used to set the hysteresis for the threshold value. This means that if the bandwidth should e.g. be 10%, the hysteresis lies between -5% and +5% at the threshold. The percentage value is therefore based on the threshold itself.

Depending on the setting “Send threshold object”, a telegram is triggered at the output communication object “if object value is out of bandwidth of hysteresis” or “on object value change”. In the first setting, a telegram is always triggered if a telegram is received at the input and the value lies outside the hysteresis limits, regardless of whether the output value has changed or not. In the second setting, an output telegram is only triggered if the output value has previously actually been changed by a new input telegram (e.g. value change from “0” to “1”).

It is defined with the corresponding parameter “If value is above the upper border of hysteresis send” or “If value is below the lower border of hysteresis send” whether an ON, OFF or no telegram should be sent when the value exceeds or falls below the hysteresis limits.

**Example of threshold values**
The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Data type of the input | **2-Byte Floating point value** |
| – Threshold value | **3** |
| – Bandwidth of the hysteresis in % of the threshold value (0…50%) | **10** |
| – Send threshold object | **if object value is out of bandwidth of hysteresis** |
| – If value is above the upper border of hysteresis send | **ON telegram** |
| – If value is below the lower border of hysteresis send | **OFF telegram** |
| – Send delay | **none** |



It is also possible to set a send delay for the send behaviour of the output. This means that the output signal is only sent once this delay has elapsed. The delay is composed of a base and a factor.

*Send delay = Base * Factor*

If the output value should change again within the send delay, the “old” output value is no longer sent. The logic module adopts the new output value and restarts the send delay.

## Scenes

With the “Scenes” function, the logic module is used as a scene module. A maximum of 8 scenes can be stored and retrieved in one function. The number of scenes that can be displayed and retrieved is dependent on the setting in the 4 parameters “Use scenes ...”. 2 scenes can be enabled via each of the 4 parameters.

The ETS program displays a common 1 bit communication object “Scene call ...” for every 2 scenes. For scenes 1 and 2 this means e.g. a telegram with the value “0” recalls scene 1 while scene 2 is recalled by a telegram with the value “1”. Any EIB switch sensor or universal interface can thus recall the stored scenes with the “Switch” application.

There are 6 output communication objects available for each scene for controlling the actuators. It is possible to control actuators with 1 bit input objects (switch or shutter actuators), actuators with 1 byte input objects (dimming actuators or 1 byte positioning for shutter actuators) or room thermostats with 2 byte input objects. The size of the output communication objects is defined separately for each output via the setting “Data type of output ...“.

| Data types: | |
|---|---|
| 1-Bit Value | EIS1 |
| 1-Byte Value in % (0…100%) | EIS6 |
| 1-Byte Value (0…255) | EIS6 |
| 2-Byte Temperature value | EIS5 |

So that the end user has the option of modifying the prestored scenes himself, there is a 1 bit communication object “Scene programming mode”. If an ON telegram is received at this communication object, all the output objects trigger read requests in sequence with the “sending” group address. The values of the received response telegrams are stored in the EEPROM, so that they are not lost after bus voltage failure.

Note:
To be able to respond to read requests, the read flag must be set for the linked communication objects of the actuators.

The end user is therefore able to set, store and retrieve his own favourite scenes himself. The following process must be noted:
– Set all the actuators or room thermostats involved to the required position or preselect the required value (via separate push buttons and group addresses)
– then activate the storing of the scene e.g. via a 1-fold switch sensor
– retrieve the scene that should be modified
– switch off the storing of the scene
– after a save which takes a certain period, he can recall the stored values at any time.

**Important:**
Unused actuator groups must have the parameter setting “no reaction”. Otherwise, it is not possible to store scenes.

The information that a scene has been stored is sent via the 1 bit communication object “Scene Save indication” after a save.

The communication object “Scene Reset to preset” is used to enable the original settings (parameter settings) of the outputs to be retrieved at any time.

The preset options of the outputs are defined separately for each output via its own parameters.

It is possible to set a “peaceful” scene structure. This is carried out via the parameter “Delay between sending the output telegrams”. No delays are set by default. This means that the output communication objects send their values directly after each other following the retrieval of a scene. In many cases, it can be more pleasant if the individual sensors are not addressed in a row. The “Delay” must be set with a base and a factor. The delay is maintained between each scene telegram of the outputs and is always the same length.

## Temperature comparator

With the function “Temperature comparator”, it is possible to compare two temperature values with each other and to trigger predefined output telegrams at specific temperature differentials.

The function “Temperature comparator” has two 2 byte input communication objects which receive temperature values. They can be determined and sent via a room thermostat or a temperature sensor.

If the logic module establishes a temperature difference, a predefined telegram is sent via the output communication object. The size of the output communication object is defined via the setting “Output data type”. The following data types can be set:

1-Byte (0…255)
1-Byte (-128…+127)
2-Byte (0…+65.535)
2-Byte (-35.768…+35.787)
1-Bit

The size of the temperature difference which should trigger an output telegram is defined via the parameter “Temperature difference in Kelvin”. Depending on the application, temperature differences of 0.5 K to 10 K (in 0.5 K steps) can be set.

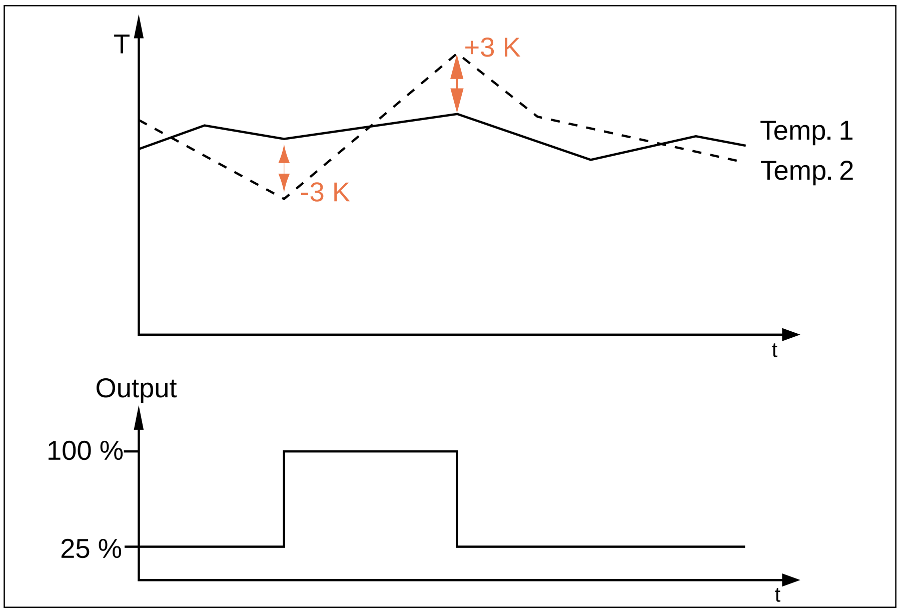
With the parameter “Send telegram if”, it is defined whether an output telegram is triggered if
– Input 1 > Input 2,
– Input 1 < Input 2 or
– Inp.1 > Inp..2 and if Inp.1 < Inp.2.

Depending on the size of the output communication object, a fixed preset value of 1 bit, 1 byte or 2 bytes can be sent. The value is set via the parameter “Send if...”.

With the parameter “Send output values”, it is defined when the output should send a telegram. This can be “on reception of new telegram” (the output is sent after receipt of an input telegram, regardless of whether the

**Example of temperature comparator**
The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Temperature difference (+/-) in Kelvin | **3** |
| – Output data type | **1-Byte (0…255)** |
| – Send telegram if | **Inp.1>Inp.2 and if Inp.1<Inp.2** |
| – Send if input 1 > input 2 | **255** |
| – Send if input 1 < input 2 | **64** |
| – Send output values | **on object value change** |
| – Send delay | **none** |



output value has changed or not) or only “on object value change” (after receipt of an input telegram that calls up a change in the output signal).

It is possible to also set a send delay in the two settings above for the send behaviour of the output. This means that the output signal is only sent once this delay has elapsed. The delay is composed of a base and a factor.
*Send delay = Base * Factor*

If the output value should change again during the send delay, the “old” output value is no longer sent. The logic module adopts the new output value and restarts the send delay.

If the output value changes several times during the send delay (setting: send output value only after a change), the current value is sent again once the send delay has elapsed.

Alternatively, the output value can be sent cyclically. To do so, the parameter “Send output values” must be set to “cyclic”. The cyclic interval with which the output value is sent is composed of a base value and a factor.
*Cyclic interval = Base * Factor*

## Gate / Filter

With the “Gate / Filter” function, it is possible to temporarily disable or enable different information flows.

Two independent gate/filter functions are available for each function. Both have their own set of parameters and communication objects.

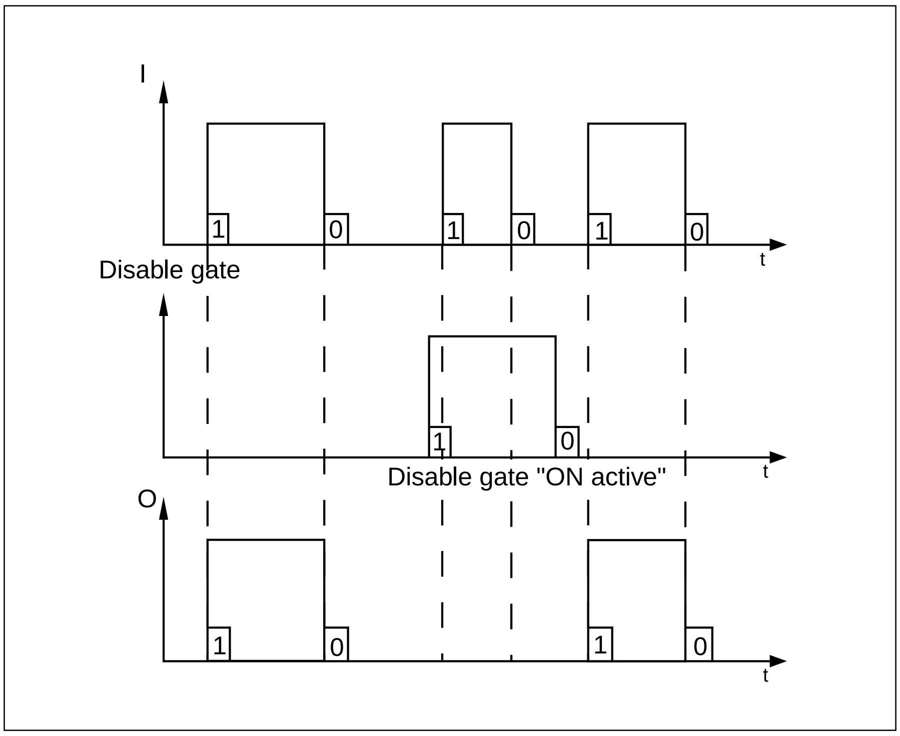
Each gate/filter has three communication objects: an input, an output and a disable object. The bit size of the input and output object is defined via the parameter “Data type”. The following data types can be selected:
1-Bit (EIS1)
2-Bit
4-Bit
1-Byte (EIS6 / EIS14)
2-Byte (EIS5 / EIS10)
4-Byte (EIS9 / EIS11)

**Note**:
The “Gate / Filter” function operates on the format level of telegrams (e.g. 1 bit, 1 byte, 2 byte) and not of EIS types.

An association between the object and an EIS type first originates from the context of the ETS project, in which this function is used.

**Example of gate/filter**
The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Data type | **1-Bit (EIS1)** |
| – Information flow | **Input to output** |
| – Filter function | **filter nothing** |
| – Output | **normal** |
| – Send output values | **on object value change** |
| – Disable gate | **if disable = 1 (ON)** |



The size of the disable object is always 1 bit. The disable object is inactive in the default setting. The object is isolated via the parameter “Disable gate”. The same parameter enables an ON or OFF telegram at the disable object to recall a disable function.
When the disable function is active, incoming telegrams at the input and output objects are not stored. This means that the last value present at the input and output object before the disable period is still active once the disable function has been cancelled.

The setting “Information flow” determines whether the telegrams (information) are routed from “Input to output”, from “Output to input“ or in “Both directions”.

If the data type of the input and output is defined as 1 bit, there are two additional parameters. It is possible to filter out ON or OFF telegrams with these parameters. The setting “Filter function” is used for this purpose. The output signal can also be inverted via the “Output” parameter.

## Staircase light

Two staircase light functions can be set for each function. There are two 1 bit communication objects for each.

The staircase light timer is triggered via the 1 bit communication object “Staircase light ... input” on receipt of an ON telegram. Once the period has elapsed, an OFF telegram is sent via the same object.

The 1 bit communication object “Staircase light ... Disable” is used to temporarily disable the staircase light function. The function is disabled if an ON telegram is sent to the object while an OFF telegram enables the function again.

If e.g. an ON telegram is received at the input object during the disable period, it is evaluated once the function has been enabled again. As soon as an OFF telegram is received at the disable object, the set delay period starts to elapse.

The staircase light delay is set using a base and factor via the corresponding parameters.
*Delay = Base * Factor*

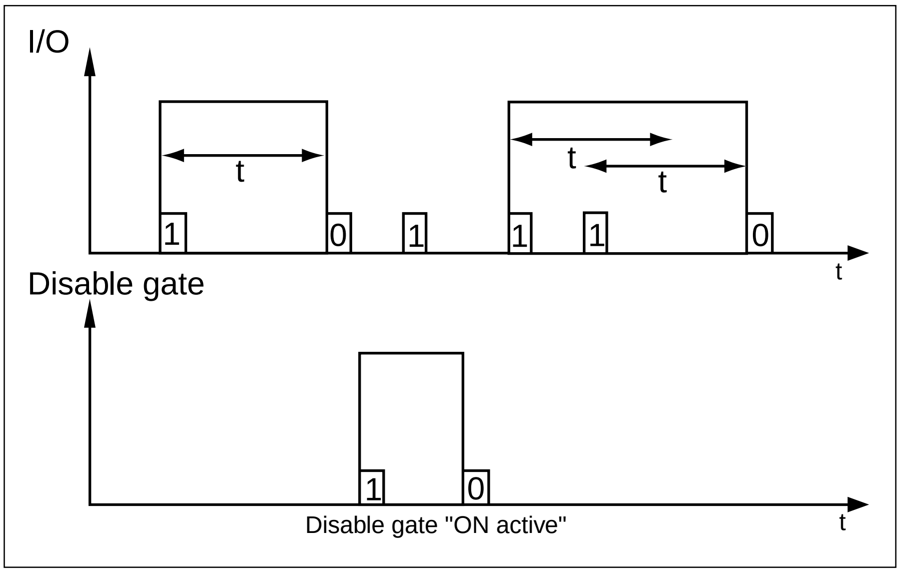
If another ON telegram is received at the input communication object during the staircase light period, the delay restarts again by default. If this behaviour should be prevented, the parameter “Staircase light is” must be set to “not retriggerable”.

## Multiplier

The “Multiplier” function is used to generate several 1 bit and 1 byte information elements from 1 bit data. The “Multiplier” function has a 1 bit “Multiplier input” communication object and up to four 1 bit “Multiplier output”communication objects as well as up to four further 1 byte “Multiplier output” communication objects.

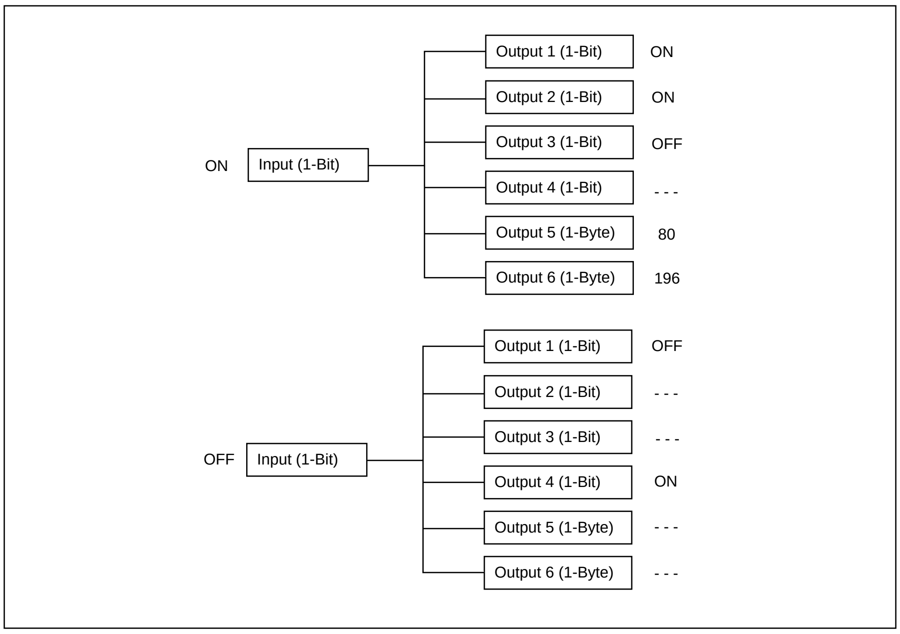
**Example of staircase light**
The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Delay | **Base: 1 min** |
| – Factor for delay (1…255) | **5** |
| – Staircase light is | **retriggerable** |



**Example of multiplier**
The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Reaction of output 1 | **like input telegram** |
| – Reaction of output 2 | **send ON if ON received** |
| – Reaction of output 3 | **send OFF if ON received** |
| – Reaction of output 4 | **send ON if OFF received** |
| – Reaction of the 1-Byte outputs | **Send value on ON telegrams** |
| – Output 5 | **used** |
| – Value | **80** |
| – Output 6 | **used** |
| – Value | **196** |
| – Output 7 | **not used** |
| – Output 8 | **not used** |



The 1 bit output objects are always available. If the 1 byte output objects should also be used, they must previously be isolated with the setting “used” via the parameter “Output ...”. Only then are the 1 byte output objects visible in ETS.

Telegrams are triggered at the output communication objects once a telegram has been received at the input communication object. This can be either an ON or OFF telegram.

It is set separately for each 1 bit output object via the parameter “Reaction of output ...” whether an ON or OFF telegram triggers a telegram at the respective output. The following options are available for selection:
– like input telegram
– like input telegram inverted
– send ON if ON received
– send OFF if ON received
– send OFF if OFF received
– send ON if OFF received

The isolated 1 byte output objects send predefined values in the event of an ON or OFF telegram at the input object. A common setting is made for all the 1 byte outputs (max. 4) with the setting “Reaction of the 1-Byte outputs” as to whether they should send a value after an ON or OFF telegram at the input.

The value that should be sent is defined individually for each output with the parameter “Value”. The value can lie between 0 and 255.

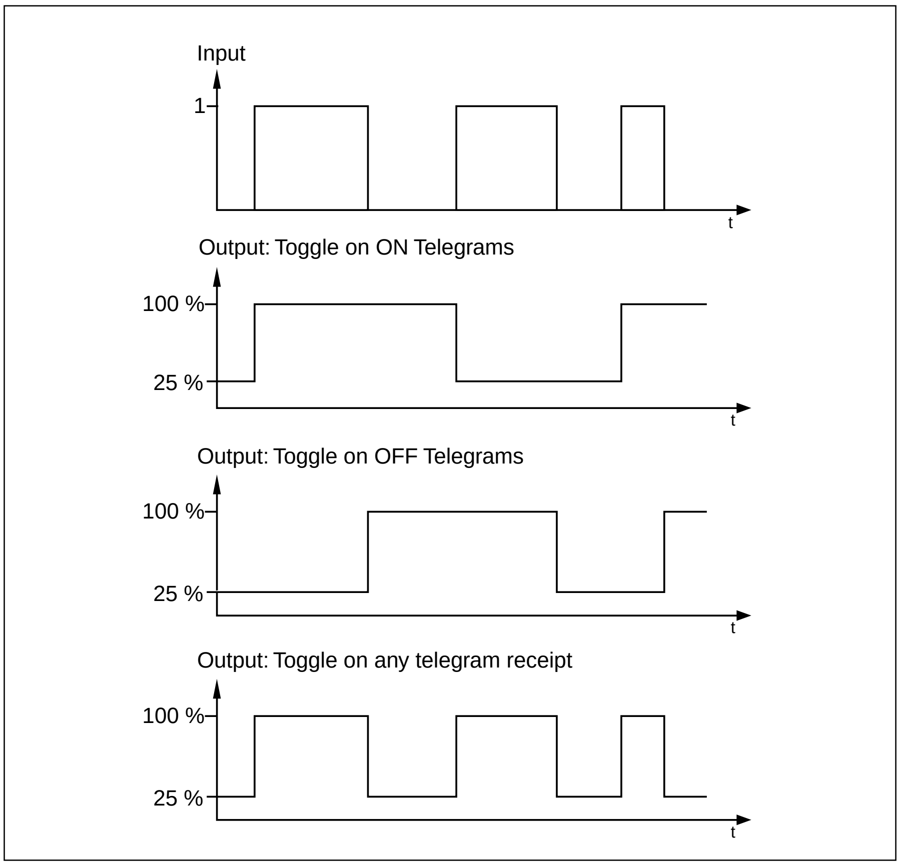
**Example of Toggle value**

The following parameter settings cause the logic module to behave in accordance with the diagram.

| | |
|---|---|
| – Output data type | **1-Byte Value in %** |
| – Setpoint value 1 | **25 %** |
| – Setpoint value 2 | **100 %** |

Three different output reactions are shown in the diagram. The title of the diagram “Output: Toggle on...” is based on the parameter

| | |
|---|---|
| – Toggle on | **(see diagram)** |



Toggle value

With the “Toggle value” function, it is possible to switch back and forth between two setpoint values. The function has five communication objects.

The setpoint values are sent via the communication object “Toggle value output”. The object can adopt the following data types:

1-Byte Value in % (EIS6)
1-Byte Orientation in degrees (EIS6)
1-Byte Counter + (EIS14)
1-Byte Counter +/- (EIS14)
2-Byte Counter + (EIS10)
2-Byte Floating point value (EIS5)
2-Byte Counter +/- (EIS10)
4-Byte Counter + (EIS11)
4-Byte Floating point value (EIS9)
4-Byte Counter +/- (EIS11)

The different data types are defined, and thereby the setpoint value sizes, via the setting “Output data type”. Setpoints 1 and 2 must be entered in the corresponding parameters.

The 1 bit object “Toggle value input” is used as an input communication object. If a telegram is received at this object, the output communication object sends the first or second setpoint value. There is thereby toggling between the first and second setpoint after each telegram at the input. This means that if setpoint 1 was sent last, setpoint 2 is sent with the subsequent telegram. If setpoint 2 was sent last, setpoint 1 is sent with the next telegram.

The parameter “Toggle on” determines whether an ON telegram, an OFF telegram or every telegram causes a change in the setpoint value. Alternatively, it can be defined with this parameter that an ON telegram at the input triggers setpoint 1 and an OFF telegram triggers setpoint 2.

The setpoint value that is currently active can be read out via the 1 bit communication object “Toggle value setpoint status”. If the object sends the value “0” then setpoint 1 is active while setpoint 2 is active if value “1” is sent.

To enable the end user to define his own setpoint, the 1 bit communication object “Toggle value save setpoints” is available. An ON telegram activates the save process while an OFF telegram deactivates it.

The 1 bit communication object “Toggle value Reset to preset” makes it possible to return to the original settings in the parameters. An ON telegram resets both setpoints to their origingal values.

It is possible to also set a send delay for the send behaviour of the output. This means that the output signal is only sent once this delay has elapsed. The delay is composed of a base and a factor.
*Send delay = Base * Factor*

## Counter

Each function of the logic module can count telegrams with the “Counter” function. Depending on the parameter “Data type of the counter”, this can be the following data types:

1-Byte (0…+255)
1-Byte (-128…+127)
2-Byte (0…+65.535)
2-Byte (-32.768…+32.767)
4-Byte (0…+4.294.967.295)
4-Byte (-2.147.483.648…
+2.147.483.647)

There are eight communication objects available for the “Counter” function. Two 1 bit communication objects are used to either count upwards or downwards by a step (“Counter up”, “Counter down”).

Via the 1 bit communication object “Counter disable”, it is possible to temporarily deactivate the counter. The disable function is switched on as soon as an ON telegram is received at the object. An OFF telegram switches the disable function off again.

The communication object “Counter value” sends the current counter content. The object size depends on the data type used (parameter: “Data type of the counter”).

The current counter content can be requested with the 1 bit communication object “Counter request values”. An ON telegram simply needs to be sent to the object. The communication object “Counter value” then sends the information.

The parameters “Count up input” and “Count down input” define whether the counter counts upwards or downwards with an ON telegram, OFF telegram or any telegram.

If there is an overflow of the counter value i.e. it reaches the end of the counting range,the counter can “revolve”, “stop” or be “reset to 0”. The parameter “Reaction on counter overflow” must be noted for the correcting mode of operation (revolve: the counter is set to the minimum value e.g. -128 or the maximum value).

The counter has an additional counter available regardless of the “Data type of the counter”. The additional counter is always 2 bytes. The current value of the additional counter is sent via the 2 byte communication object “Counter add. counter value”.

The additional counter also has the 1 bit communication objects “Counter add. counter overflow” and “Counter reset add. counter”.

The “Counter add. counter overflow” object sends an ON telegram in the event of an overflow. The additional counter can revert to its preset value via the “Counter reset add. counter” object. This value is defined with the parameter “Preset value for over/ underflow”.

This value can be changed independently by the user. To do so, the setting “Preset value can be overwritten” must be set to “yes”. If an ON telegram is now received at the “Counter reset add. counter” object, the additional counter adopts the current value as the preset value.

**Bus voltage failure/recovery**

In the event of a bus voltage failure, the current counter values are not lost. They are written by the RAM into the EEPROM when the bus voltage fails.

On bus voltage recovery, the counter value can be sent automatically. To do so, the corresponding parameter “Send counter values on bus voltage recovery” must be set to “Yes”.

## Time delay

ON and OFF delays are implemented with the “Time delay” function. Four independent time delays can be set for each function. They all have their own set of parameters and communication objects.

The time delay function has three communication objects: an input, an output and a disable object.

The size of the input and output object depends on the setting in the parameter “Time delay .... - data type”. This can be one of the following data types:

| |
|---|
| 1-Bit (EIS1) |
| 2-Bit |
| 4-Bit |
| 1-Byte (EIS6 / EIS14) |
| 2-Byte (EIS5 / EIS10) |
| 4-Byte (EIS9 / EIS11) |
| |

The 1 bit disable object enables the function to be deactivated via an ON telegram. An OFF telegram reactivates the function.

When the function is disabled, the received telegrams are not evaluated.

The delay periods are defined with a base and a factor.
*Delay = Base * Factor*

If “1-Bit” is selected as the data type, it is possible to implement an “ON delay”, an “OFF delay” and an “ON and OFF delay” via the parameter “Function”. The delay periods are defined with a base and a factor. If e.g. the setting “ON and OFF delay” is selected, the same times apply for the ON and OFF delay. For all other data types, the time delay acts as a telegram delay. This means that when a telegram is received at the input, the delay period is started and the telegram is only sent to the output once the period has elapsed. If a further telegram with a different value is received during the delay, this new value is sent at the end of the delay period.

Depending on the parameter “Time delay is”, the ON or OFF delay is “retriggerable” or “not retriggerable”. If the function can be retriggered, it means that an input telegram during an active delay period restarts the delay.

**Communication objects**
Format converter
**2x1 Bit --> 2 Bit**
for function A, B and C.

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Format converter input Bit 1 | Receive telegram |
| 1 | 1 bit | A: Format converter input Bit 2 | Receive telegram |
| 8 | 2 bit | A: Format converter output 2-Bit | Send telegram |
| 16 | 1 bit | B: Format converter input Bit 1 | Receive telegram |
| 17 | 1 bit | B: Format converter input Bit 2 | Receive telegram |
| 24 | 2 bit | B: Format converter output 2-Bit | Send telegram |
| 32 | 1 bit | C: Format converter input Bit 1 | Receive telegram |
| 33 | 1 bit | C: Format converter input Bit 2 | Receive telegram |
| 40 | 2 bit | C: Format converter output 2-Bit | Send telegram |

**Note:**
For improved clarity, only the communication objects for function A are listed in the following section. The functions B and C have the same communication objects when they have the appropriate parameter settings.

**Communication objects**
Format converter
**8x1 Bit --> 1 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Format converter input Bit 1 | Receive telegram |
| 1 | 1 bit | A: Format converter input Bit 2 | Receive telegram |
| 2 | 1 bit | A: Format converter input Bit 3 | Receive telegram |
| 3 | 1 bit | A: Format converter input Bit 4 | Receive telegram |
| 4 | 1 bit | A: Format converter input Bit 5 | Receive telegram |
| 5 | 1 bit | A: Format converter input Bit 6 | Receive telegram |
| 6 | 1 bit | A: Format converter input Bit 7 | Receive telegram |
| 7 | 1 bit | A: Format converter input Bit 8 | Receive telegram |
| 8 | 1 byte | A: Format converter output Byte | Send telegram |

**Communication objects**
Format converter
**1 Byte --> 2 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 byte | A: Format converter input Byte | Receive telegram |
| 8 | 2 byte | A: Format converter output 2-Byte | Send telegram |

**Communication objects**
Format converter
**2x1 Byte --> 2 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 byte | A: Format converter input Low Byte | Receive telegram |
| 1 | 1 byte | A: Format converter input High Byte | Receive telegram |
| 8 | 2 byte | A: Format converter output 2-Byte | Send telegram |

**Communication objects**
Format converter
**2x2 Byte --> 4 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: Format converter input 2-Byte #1 | Receive telegram |
| 1 | 2 byte | A: Format converter input 2-Byte #2 | Receive telegram |
| 8 | 4 byte | A: Format converter output 4-Byte | Send telegram |

**Communication objects**
Format converter
**2 Bit --> 2x1 Bit**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 bit | A: Format converter input 2-Bit | Receive telegram |
| 8 | 1 bit | A: Format converter output 1-Bit 1 | Send telegram |
| 9 | 1 bit | A: Format converter output 1-Bit 2 | Send telegram |

**Communication objects**
Format converter
**1 Byte --> 8x1 Bit**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 byte | A: Format converter input Byte | Receive telegram |
| 8 | 1 bit | A: Format converter output Bit 1 | Send telegram |
| 9 | 1 bit | A: Format converter output Bit 2 | Send telegram |
| 10 | 1 bit | A: Format converter output Bit 3 | Send telegram |
| 11 | 1 bit | A: Format converter output Bit 4 | Send telegram |
| 12 | 1 bit | A: Format converter output Bit 5 | Send telegram |
| 13 | 1 bit | A: Format converter output Bit 6 | Send telegram |
| 14 | 1 bit | A: Format converter output Bit 7 | Send telegram |
| 15 | 1 bit | A: Format converter output Bit 8 | Send telegram |

**Communication objects**
Format converter
**2 Byte --> 2x1 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: Format converter input 2-Byte | Receive telegram |
| 8 | 1 byte | A: Format converter output Low Byte | Send telegram |
| 9 | 1 byte | A: Format converter output High Byte | Send telegram |

**Communication objects**
Format converter
**4 Byte --> 2x2 Byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 4 byte | A: Format converter input 4-Byte | Receive telegram |
| 8 | 2 byte | A: Format converter output 2-Byte #1 | Send telegram |
| 9 | 2 byte | A: Format converter output 2-Byte #2 | Send telegram |

**Communication objects**
Logic

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Logic input 1 | Receive telegram |
| 1 | 1 bit | A: Logic input 2 | Receive telegram |
| 2 | 1 bit | A: Logic input 3 | Receive telegram |
| 3 | 1 bit | A: Logic input 4 | Receive telegram |
| 4 | 1 bit | A: Logic input 5 | Receive telegram |
| 5 | 1 bit | A: Logic input 6 | Receive telegram |
| 6 | 1 bit | A: Logic input 7 | Receive telegram |
| 7 | 1 bit | A: Logic input 8 | Receive telegram |
| 8 | 1 bit | A: Logic output | Send telegram |

**Communication objects**
Min./Max. Value indication
**1 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 byte | A: MinMax input 1 | Receive telegram |
| 1 | 1 byte | A: MinMax input 2 | Receive telegram |
| 2 | 1 byte | A: MinMax input 3 | Receive telegram |
| 3 | 1 byte | A: MinMax input 4 | Receive telegram |
| 4 | 1 byte | A: MinMax output | Send telegram |

**Communication objects**
Min./Max. Value indication
**2 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: MinMax input 1 | Receive telegram |
| 1 | 2 byte | A: MinMax input 2 | Receive telegram |
| 2 | 2 byte | A: MinMax input 3 | Receive telegram |
| 3 | 2 byte | A: MinMax input 4 | Receive telegram |
| 4 | 2 byte | A: MinMax output | Send telegram |

**Communication objects**
Min./Max. Value indication
**4 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 4 byte | A: MinMax input 1 | Receive telegram |
| 1 | 4 byte | A: MinMax input 2 | Receive telegram |
| 2 | 4 byte | A: MinMax input 3 | Receive telegram |
| 3 | 4 byte | A: MinMax input 4 | Receive telegram |
| 4 | 4 byte | A: MinMax output | Send telegram |

**Communication objects**
Threshold values
**1 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 byte | A: Threshold value input 1 | Receive telegram |
| 1 | 1 bit | A: Threshold value T. value 1 | Send telegram |
| 2 | 1 byte | A: Threshold value input 2 | Receive telegram |
| 3 | 1 bit | A: Threshold value T. value 2 | Send telegram |

**Communication objects**
Threshold values
**2 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: Threshold value input 1 | Receive telegram |
| 1 | 1 bit | A: Threshold value T. value 1 | Send telegram |
| 2 | 2 byte | A: Threshold value input 2 | Receive telegram |
| 3 | 1 bit | A: Threshold value T. value 2 | Send telegram |

**Communication objects**
Threshold values
**4 byte**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 4 byte | A: Threshold value input 1 | Receive telegram |
| 1 | 1 bit | A: Threshold value T. value 1 | Send telegram |
| 2 | 4 byte | A: Threshold value input 2 | Receive telegram |
| 3 | 1 bit | A: Threshold value T. value 2 | Send telegram |

**Communication objects**
Scenes
**Here: 1 bit**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Scene output 1 | Send telegram |
| 1 | 1 bit | A: Scene output 2 | Send telegram |
| 2 | 1 bit | A: Scene output 3 | Send telegram |
| 3 | 1 bit | A: Scene output 4 | Send telegram |
| 4 | 1 bit | A: Scene output 5 | Send telegram |
| 5 | 1 bit | A: Scene output 6 | Send telegram |
| 6 | 1 bit | A: Scene call S1S2 | OFF = Sc 1, ON = Sc 2 |
| 7 | 1 bit | A: Scene call S3S4 | OFF = Sc 3, ON = Sc 4 |
| 8 | 1 bit | A: Scene call S5S6 | OFF = Sc 5, ON = Sc 6 |
| 9 | 1 bit | A: Scene call S7S8 | OFF = Sc 7, ON = Sc 8 |
| 10 | 1 bit | A: Scene Programming mode | Programming mode on/off |
| 11 | 1 bit | A: Scene Save indication | Scene was stored |
| 12 | 1 bit | A: Scene Reset to preset | Receive telegram |

**Communication objects**
Temperature comparator
**1 bit output**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: T. Comparator input 1 | Input |
| 1 | 2 byte | A: T. Comparator input 2 | Input |
| 2 | 1 bit | A: T. Comparator output | Output |

**Communication objects**
Temperature comparator
**1 byte output**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: T. Comparator input 1 | Input |
| 1 | 2 byte | A: T. Comparator input 2 | Input |
| 2 | 1 byte | A: T. Comparator output | Output |

**Communication objects**
Temperature comparator
**2 byte output**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 2 byte | A: T. Comparator input 1 | Input |
| 1 | 2 byte | A: T. Comparator input 2 | Input |
| 2 | 2 byte | A: T. Comparator output | Output |

**Communication objects**
Gate / Filter
**Here: 1 bit**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Gate 1 input | Receive telegram |
| 1 | 1 bit | A: Gate 1 disable | Receive telegram |
| 2 | 1 bit | A: Gate 1 output | Send telegram |
| 3 | 1 bit | A: Gate 2 input | Receive telegram |
| 4 | 1 bit | A: Gate 2 disable | Receive telegram |
| 5 | 1 bit | A: Gate 2 output | Send telegram |

**Communication objects**
Staircase light

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Staircase light 1 input | Send/receive telegram |
| 1 | 1 bit | A: Staircase light 1 disable | Receive telegram |
| 2 | 1 bit | A: Staircase light 2 input | Send/receive telegram |
| 3 | 1 bit | A: Staircase light 2 disable | Receive telegram |

**Communication objects**
Multiplier

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Multiplier input | Receive telegram |
| 1 | 1 bit | A: Multiplier output 1 | Send telegram |
| 2 | 1 bit | A: Multiplier output 2 | Send telegram |
| 3 | 1 bit | A: Multiplier output 3 | Send telegram |
| 4 | 1 bit | A: Multiplier output 4 | Send telegram |
| 5 | 1 byte | A: Multiplier output 5 | Send telegram |
| 6 | 1 byte | A: Multiplier output 6 | Send telegram |
| 7 | 1 byte | A: Multiplier output 7 | Send telegram |
| 8 | 1 byte | A: Multiplier output 8 | Send telegram |

**Communication objects**
Toggle Value
**Here: 1 byte output**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Toggle value input | Receive telegram |
| 3 | 1 bit | A: Toggle value save setpoints | Receive telegram |
| 4 | 1 bit | A: Toggle value Reset to preset | Receive telegram |
| 5 | 1 byte | A: Toggle value output | Send telegram |
| 6 | 1 bit | A: Toggle value setpoint status | Send telegram |

**Communication objects**
Counter
**Here: 1 byte output**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Counter up | Receive telegram |
| 1 | 1 bit | A: Counter down | Receive telegram |
| 2 | 1 bit | A: Counter disable | Receive telegram |
| 3 | 1 bit | A: Counter request values | Receive telegram |
| 4 | 1 byte | A: Counter value | Send telegram |
| 5 | 2 byte | A: Counter add. counter value | Send telegram |
| 6 | 1 bit | A: Counter add. counter overflow | Send telegram |
| 7 | 1 bit | A: Counter reset add. counter | Receive telegram |

**Communication objects**
Time delay
**Here: 1 bit**

| No. | Type | Object name | Function |
|---|---|---|---|
| 0 | 1 bit | A: Time delay 1 input | Receive telegram |
| 1 | 1 bit | A: Time delay 1 disable | Receive telegram |
| 2 | 1 bit | A: Time delay 1 output | Send telegram |
| 3 | 1 bit | A: Time delay 2 input | Receive telegram |
| 4 | 1 bit | A: Time delay 2 disable | Receive telegram |
| 5 | 1 bit | A: Time delay 2 output | Send telegram |
| 6 | 1 bit | A: Time delay 3 input | Receive telegram |
| 7 | 1 bit | A: Time delay 3 disable | Receive telegram |
| 8 | 1 bit | A: Time delay 3 output | Send telegram |
| 9 | 1 bit | A: Time delay 4 input | Receive telegram |
| 10 | 1 bit | A: Time delay 4 disable | Receive telegram |
| 11 | 1 bit | A: Time delay 4 output | Send telegram |

**Parameters**
The default setting for the values is **printed in bold type.**

| | |
|---|---|
| General: | |
| Separate for each function: | |
| – Function … | **not used** |
| | Format converter |
| | Logic |
| | Min./Max. Value indication |
| | Threshold values |
| | Scenes |
| | Temperature comparator |
| | Gate / Filter |
| | Staircase light |
| | Multiplier |
| | Toggle Value |
| | Counter |
| | Time delay |

**Parameters**
with **Format converter** function.

The default setting for the values is **printed in bold type.**

9

| | | |
|---|---|---|
| Format converter: | | |
| … - Format converter: | | |
| – Converter Function | 2x1-Bit | --> 1x2-Bit |
| | 8x1-Bit | --> 1-Byte |
| | 1x1-Byte | --> 2-Byte |
| | 2x1-Byte | --> 2-Byte |
| | 2x2-Byte | --> 4-Byte |
| | 1x2-Bit | --> 2x1-Bit |
| | **1-Byte** | **--> 8x1-Bit** |
| | 2-Byte | --> 2x1-Byte |
| | 4-Byte | --> 2x2-Byte |
| Only for 1x1-Byte --> 2-Byte: | | |
| --> Note: High byte of the output = 0<br>Low byte of the output = input byte | | |
| – Read input values over EIB after bus voltage recovery | Yes / **No** | |
| – Send output values | **on reception of new telegram**<br>on object value change | |

**Parameter**
with **Logic** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Logic: | |
| … - Logic Inp. 1-4: | |
| – Logical operation | **AND** |
| | OR |
| | Exclusive OR |
| | NAND |
| | NOR |
| | Equivalence |
| Separate for each input: | |
| – Input… | not used |
| | **normal** |
| | inverted |
| Only when input is used: | |
| – Object value after bus voltage recovery | read values over EIB |
| | **use value before bus voltage fail** |
| | use 0 as preset |
| | use 1 as preset |
| … - Logic Inp. 5-8: | |
| Separate for each input: | |
| – Input … | **not used** |
| | normal |
| | inverted |
| Only when input is used: | |
| – Object value after bus voltage recovery | read values over EIB |
| | **use value before bus voltage fail** |
| | use 0 as preset |
| | use 1 as preset |
| … - Logic output: | |
| – Output | **normal** |
| | inverted |
| – Send output values | **on reception of new telegram** |
| | on object value change |
| | cyclic |
| Only if cyclical sending is selected: | |
| – Send delay | **none** |
| | Base: 100 ms |
| | Base: 1 s |
| | Base: 10 s |
| | Base: 1 min |
| | Base: 10 min |
| | Base: 1 h |
| Only if send delay is selected: | |
| – Send delay factor (1…255) | **1** |
| Only if cyclical sending is selected: | |
| – Cyclic sending base | 1 s / 10 s / **1 min** / 10 min / 1 h |
| – Cyclic sending factor | **1** |

**Parameters**
with **Min./Max. Value indication** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Min./Max. Value indication: | |
| … - Min./Max. Value indication: | |
| – Data type of the inputs | 1-Byte Value in % [0…100%] (EIS6)<br>1-Byte Orientation in degrees [0…360](EIS6)<br>**1-Byte Counter [0…+255] (EIS14)**<br>1-Byte Counter [-128…+127] (EIS14)<br>2-Byte Counter [0…+65.535] (EIS10)<br>2-Byte Floating point value (EIS5)<br>2-Byte Counter [-32.768…+32.767] (EIS10)<br>4-Byte Floating point value (EIS9)<br>4-Byte Counter [-10.000.000…+10.000.000] (EIS11) |
| Separate for each input: | |
| – Input … | not used<br>**used** |
| – Output | **takes the highest value**<br>takes the lowest value<br>takes the average value |
| – Send output values | **on reception of new telegram**<br>on object value change<br>cyclic |
| Only if cyclical sending is selected: | |
| – Cyclic sending base | 1 s / 10 s / **1 min** / 10 min / 1 h |
| – Cyclic sending factor | **1** |

**Parameters**
with **Threshold values** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Threshold values: | |
| Separate for each threshold: | |
| … - Threshold value …: | |
| – Data type of the input | 1-Byte Value in % [0…100%] (EIS6)<br>1-Byte Orientation in degrees [0…360](EIS6)<br>**1-Byte Counter [0…+255] (EIS14)**<br>1-Byte Counter [-128…+127] (EIS14)<br>2-Byte Counter [0…+65.535] (EIS10)<br>2-Byte Floating point value (EIS5)<br>2-Byte Counter [-32.768…+32.767] (EIS10)<br>4-Byte Floating point value (EIS9)<br>4-Byte Counter [-10.000.000…+10.000.000] (EIS11) |
| Only for "1-Byte Value in %": | |
| – Threshold value [25=10%, 255 =10%] | **0** |
| Only for "1-Byte Orientation in degrees": | |
| – Threshold value [255 = 360°] | **0** |
| Only for "4-Byte Counter": | |
| – Threshold value (+/-1.000.000.000) | **0** |
| For all other data types: | |
| – Threshold value | **0** |
| – Bandwidth of the hysteresis in % of the threshold value [0…50%] | **0** |
| – Send threshold object | **if object value is out of bandwidth of hysteresis**<br>on object value change |
| – If value is above the upper border of hysteresis send | OFF telegram<br>**ON telegram**<br>no telegram |
| – If value is below the lower border of hysteresis send | OFF telegram<br>**ON telegram**<br>no telegram |
| – Send delay | **none**<br>Base: 100 ms<br>Base: 1 s<br>Base: 10 s<br>Base: 1 min<br>Base: 10 min<br>Base: 1 h |
| Only if send delay is selected: | |
| – Send delay factor (1…255) | **1** |

**Parameters**
with **Scenes** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Scenes: | |
| … - Scenes Common | |
| – Use scenes 1/2 | **Yes** / No |
| – Use scenes 3/4 | **Yes** / No |
| – Use scenes 5/6 | **Yes** / No |
| – Use scenes 7/8 | **Yes** / No |
| – Reset object is | **not used** |
| | used |
| | |
| … - Output types: | |
| Separate for each output: | |
| – Data type of output … | 1-Bit Value (EIS1) |
| | **1-Byte Value in % [0…100%] (EIS6)** |
| | 1-Byte Value [0…255] |
| | 2-Byte Temperature value (EIS5) |
| | |
| Separate for each scene: | |
| … - Scene …: | |
| Separate for each output: | |
| Only for “1 Bit Value”: | |
| – Preset output … | OFF |
| | **ON** |
| | no reaction |
| Only for “1-Byte Value in %”: | |
| – Preset output … | no reaction |
| | 0 % |
| | **10 %** |
| | … |
| | 90 % |
| | 100 % |
| Only for “1-Byte Value”: | |
| – Preset output … | **1** |
| Only for “2-Byte Temperature value”: | |
| – Preset output … | **1** |
| | |
| – Delay between sending the output telegrams | **none** |
| | Base: 100 ms |
| | Base: 1 s |
| | Base: 10 s |
| | Base: 1 min |
| | Base: 10 min |
| | Base: 1 h |
| Only if delay is selected: | |
| – Factor [1…255] | **1** |

**Parameters**
with **Temperature comparator** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Temperature comparator: | |
| … - Temperature comparator: | |
| – Temperature difference (+/-) in Kelvin | 0.5 / **1.0** / 1.5 / … / 9.5 / 10 |
| – Output data type | 1-Byte [0…+255]<br>1-Byte [-128…+127]<br>2-Byte [0…+65.535]<br>2-Byte [-32.768…+32.787]<br>**1-Bit** |
| – Send telegram if | Input 1 > Input 2<br>Input 1 < Input 2<br>**Inp. 1 > Inp. 2 and if Inp. 1 < Inp. 2** |
| Only for "1-Byte [0…255]": | |
| – Send if input 1 > input 2 | **0** |
| – Send if input 1 < input 2 | **255** |
| Only for "1-Byte [-128…+127]": | |
| – Send if input 1 > input 2 | **0** |
| – Send if input 1 < input 2 | **0** |
| Only for "2-Byte [0…+65.535]": | |
| – Send if input 1 > input 2 | **0** |
| – Send if input 1 < input 2 | **65535** |
| Only for "2-Byte [-32.768…+32.787]": | |
| – Send if input 1 > input 2 | **32767** |
| – Send if input 1 < input 2 | **0** |
| Only for "1-Bit": | |
| – Send if input 1 > input 2 | **ON telegram**<br>OFF telegram |
| – Send if input 1 < input 2 | **255** |
| – Send output values | **on reception of new telegram**<br>on object value change<br>cyclic |
| Only for cyclical sending: | |
| – Cyclic sending base | 1 s / 10 s / **1 min** / 10 min / 1 h |
| – Cyclic sending factor | **1** |
| Only if cyclical sending is not selected: | |
| – Send delay | **none**<br>Base: 100 ms<br>Base: 1 s<br>Base: 10 s<br>Base: 1 min<br>Base: 10 min<br>Base: 1 h |
| Only for send delay: | |
| – Send delay factor<br>(1…255) | **1** |

**Parameters**
with **Gate / Filter** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Gate / Filter: | |
| Separate for each gate/filter: | |
| … - Gate/Filter …: | |
| – Data type | **1-Bit (EIS1)**<br>2-Bit<br>4-Bit<br>1-Byte (EIS6 / EIS14)<br>2-Byte (EIS5 / EIS10)<br>4-Byte (EIS9 / EIS11) |
| – Information flow | **Input to output**<br>Output to input<br>Both directions |
| Only for “1-Bit”: | |
| – Filter function | **filter nothing**<br>filter ON telegrams<br>filter OFF telegrams |
| – Output | **normal**<br>inverted |
| – Send output values | **on reception of new telegram**<br>on object value change |
| – Disable gate | **No**<br>if disable = 1 (ON)<br>if disable = 0 (OFF) |
| Only if gate is disabled: | |
| – Note: During disable the input value is not saved | |

**Parameters**
with **Staircase light** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Staircase light: | |
| … - Staircase light 1+2: | |
| Separate for each staircase light: | |
| Staircase light …: | |
| – Delay | Base: 100 ms<br>Base: 1 s<br>Base: 10 s<br>**Base: 1 min**<br>Base: 10 min<br>Base: 1 h |
| – Factor for delay [1…255] | **1** |
| – Staircase light is | **retriggerable**<br>not retriggerable |

**Parameters**
with **Multiplier** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Multiplier: | |
| … - Multiplier 1-Bit: | |
| Separate for outputs 1 to 4: | |
| – Reaction of output … | **like input telegram**<br>like input telegram inverted<br>send ON if ON received<br>send OFF if ON received<br>send OFF if OFF received<br>send OFF if ON received |
| … - Multiplier 1-Byte: | |
| – Reaction of the 1-Byte outputs | **Send value on ON telegrams**<br>Send value on OFF telegrams |
| Separate for outputs 5 to 8: | |
| – Output… | used<br>**not used** |
| Only if used: | |
| – Value [0…255] | **0** |

**Parameters**
with **Toggle Value** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Toggle Value: | |
| … - Toggle value: | |
| – Output data type | 1-Byte Value in % (EIS6)<br>1-Byte Orientation in degrees (EIS6)<br>**1-Byte Counter + (EIS14)**<br>1-Byte Counter +/- (EIS14)<br>2-Byte Counter + (EIS10)<br>2-Byte Floating point value (EIS5)<br>2-Byte Counter +/- (EIS10)<br>4-Byte Counter + (EIS11)<br>4-Byte Floating point value (EIS9)<br>4-Byte Counter +/- (EIS11) |
| Only for “1-Byte Value in %”: | |
| – Setpoint value 1 [25=10%, 255=100%] | **0** |
| – Setpoint value 1 is | **fix**<br>rewritable over EIB |
| – Setpoint value 2 [25=10%, 255=100%] | **0** |
| – Setpoint value 2 is | **fix**<br>rewritable over EIB |
| Only for “1-Byte Orientation in degrees”: | |
| – Setpoint value 1 [7=10 Grad, 255=360 Grad] | **0** |
| – Setpoint value 1 is | **fix**<br>rewritable over EIB |
| – Setpoint value 2 [7=10 Grad, 255=360 Grad] | **0** |
| – Setpoint value 2 is | **fix**<br>rewritable over EIB |
| For all other data types: | |
| – Setpoint value 1 | **0** |
| – Setpoint value 1 is | **fix**<br>rewritable over EIB |
| – Setpoint value 2 | **0** |
| – Setpoint value 2 is | **fix**<br>rewritable over EIB |
| – Toggle on | **any telegram receipt**<br>ON telegrams only<br>OFF telegrams only<br>ON telegr. to value 1 /<br>OFF telegr. to value 2 |
| – Send delay | **none**<br>Base: 100 ms<br>Base: 1 s<br>Base: 10 s<br>Base: 1 min<br>Base: 10 min<br>Base: 1 h |
| Only for send delay: | |
| – Send delay factor (1…255) | **1** |

**Parameters**
with **Counter** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Counter: | |
| … - Counter: | |
| – Data type of the counter | 1-Byte [0…+255]<br>1-Byte [-128…+127]<br>**2-Byte [0…+65.535]**<br>2-Byte [-32.768…+32.787]<br>4-Byte [0…4.294.967.295]<br>4-Byte [-2.147.483.648…<br>+2.147.483.647] |
| – Count up input | react only on OFF telegram<br>**react only on ON telegram**<br>react on any telegram |
| – Count down input | react only on OFF telegram<br>**react only on ON telegram**<br>react on any telegram |
| – Reaction on counter overflow | **revolve**<br>stop<br>reset to 0 |
| – Send counter values on bus voltage recovery | Yes / **No** |
| – Use additional counter | Yes / **No** |
| – Send cyclic counter values | Yes / **No** |
| Only for cyclical sending: | |
| – Base for cyclic sending | 1 s<br>10 s<br>**1 min**<br>10 min<br>1 h |
| – Factor for cyclic sending | **1** |
| | |
| Only if additional counter is used: | |
| – Preset value for over/underflow (-32.768…32.768) | **0** |
| – Preset value can be overwritten | Yes / **No** |
| | |

**Parameters**
with **Time delay** function.

The default setting for the values is **printed in bold type.**

| | |
|---|---|
| Time delay: | |
| Separate for each time delay: | |
| … - Time delay …: | |
| – Time delay … - data type | **1-Bit Value (EIS1)**<br>2-Bit Value<br>4-Bit Value<br>1-Byte Value (EIS6) / (EIS14)<br>2-Byte Value (EIS5) / (EIS10)<br>4-Byte Value (EIS9) / (EIS11) |
| Only for “1-Bit Value” data type: | |
| – Function | OFF delay<br>**ON delay**<br>ON and OFF delay |
| For all other data types: | |
| – Function | **Telegram delay** |
| – Base for delay | 1 s<br>10 s<br>**1 min**<br>10 min<br>1 h |
| – Factor for delay | **1** |
| – Time delay is | **retriggerable**<br>not retriggerable |