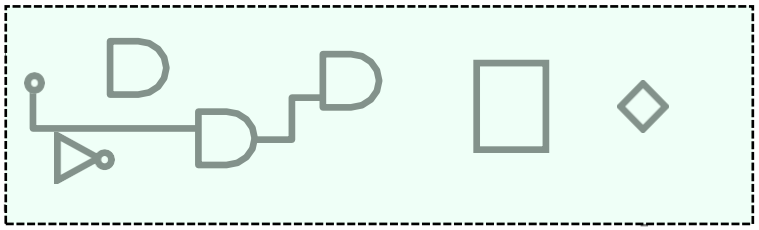
Zennio
Passion for KNX innovation



Logische Funktionen

Logisches-Mathematik-Modul

Benutzerhandbu

Handbuch Version: [0.3]_a

www.zennio.com

Inhalt

Dokument Aktualisierungen

Version	Änderungen	Seite(n)
[0.3]_a	Software Änderungen: • Interne Verzögerung bei simultanem Senden. • Zyklisches und verzögertes Senden und verzögertes Senden vorheriger Werte werden abgebrochen bei erneuter Auslösung der Funktion.	-
	Zyklisches und verzögertes Senden und verzögertes Senden vorheriger Werte werden abgebrochen bei erneuter Auslösung der Funktion.	17, 18
[0.2]_b	Erklärung über fortlaufende Funktionsauslösung mit verzögertem Ergebnis	18
	Umwandlung zu ein-Byte Prozentwerten sind gleich zu Umwandlungen zu ein-Byte Ganzzahl-Werten (es gibt keinen extra-Umwandlungstypen).	22, 23

1 EINLEITUNG

1.1 Logikfunktions-Modul

Eine große Zahl unterschiedlicher Zennio Geräte (etwa alle Aktoren der ACTinBOX und MAXinBOX Reihe) besitzen ein integriertes Logikmodul, mit dem **mathematische** und **binäre Logiken** mit Werten vom KX Bus ausgeführt und die Ergebnisse über spezielle Kommunikationsobjekte auf den Bus gesendet werden können.

Die Operanden dieser Funktionen bestehen aus den folgenden Typen:

- **Kommunikationsobjekte,** empfangen vom KNX Bus.
- **Internen Variablen,** die Teilergebnisse aus vorangegangenen Operationen beinhalten.
- **Konstante Werte**, definiert in ETS Parametern.

Bis zu zehn unterschiedliche und unabhängige Funktionen können konfiguriert werden, jede dieser Operationen kann aus bis zu 4 aufeinander folgenden Operationen bestehen, diese können interne Variablen miteinander teilen, so dass etwa ein Ergebnis als Eingangsobjekt für die nächste Operation verwendet wird.

Wichtig: *Abhängig vom entsprechenden Gerät kann die Zahl der verfügbaren Logikfunktionen variieren. Aus diesem Grund wurde das Handbuch für jedes Gerät entsprechend angepasst. Bitte nutzen Sie die Download-Links auf unserer Website(www.zennio.com)*
Um die gerätespezifische Dokumentation einzusehen.

2 KONFIGURATION

2.1 ALLGEMEINES VORGEHEN

Das Logikmodul ermöglicht das Aktivieren und Konfigurieren von bis zu zehn unabhängigen Logikfunktionen, die typischerweise in vier Abschnitte unterteilt sind:

- **Trigger (Auslöser)**: der erste Schritt ist die Auslösung, das Aufrufen der Funktion. Ein oder mehr Kommunikationsobjekte können so konfiguriert werden, das wenn eines seinen Wert via KNX Bus aktualisiert, die Funktion ausgelöst wird (wenn die Bedingung zur Ausführung zu TRUE evaluiert).

- **Execution Condition (Bedingung zur Ausführung)**: nach Auslösen der Funktion wird die Bedingung zur Ausführung geprüft (True oder False). Diese Bedingung besteht aus dem Vergleich zweier Operanden. Ist der Vergleich wahr (True), wird die Operation ausgeführt. Wurde keine Bedingung zur Ausführung konfiguriert, wird die Funktion immer bei Auslösung ausgeführt.

- **Operations (Operationen)**: das Auslösen der Funktion führt direkt zur Ausführung von bis zu vier mathematischen oder binären Operationen. Die folgenden Parameter müssen für jede Operation parametriert werden:

  - **Operation type (Typ)**: gewünschte Aktion (Addition, Subtraktion, Negation, etc.).

  - **Operands (Operanden)**: Werte für die Operation. Dies können Eingangs-Kommunikationsobjekte, interne Variablen mit Ergebnissen vorangegangener Operationen sein oder konstante Werte, die via ETS vorgegeben werden.

  - **Result (Ergebnis)**: interne Variable, in der das Ergebnis der Operation gespeichert wird.

- **Result Ergebnis)**: es ist notwendig, festzulegen, welche Variable das Endergebnis der Funktion speichert, so dass das dieser Wert über das korrespondierende Kommunikationsobjekt nach Ausführung aller Operationen gesendet werden kann.

Abb. 1 zeigt in Diagrammübersicht die Operationen der logischen Funktionen.

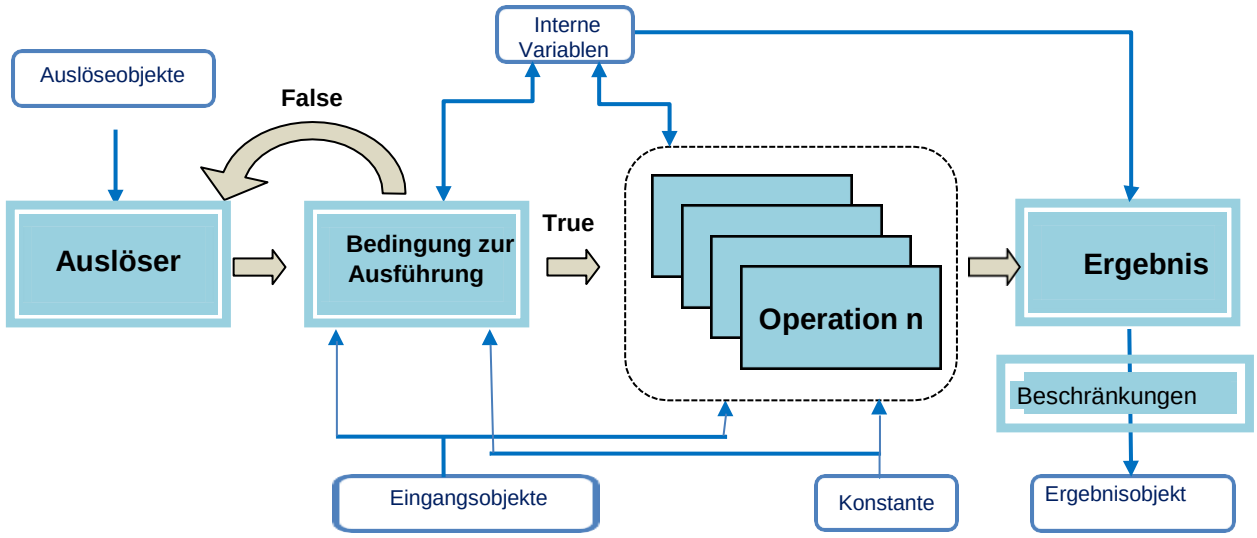


Abb. 1. Diagrammübersicht logische Operationen

2.2 AUSLÖSER

Bis zu 8 Auslöseobjekte (ein-Bit, ein-Byte, zwei-Byte oder vier-Byte) können pro Funktion festgelegt werden, jedes davon löst die Funktion aus, sobald der entsprechende Wert via KNX Bus empfangen wird. Diese Objekte müssen nicht zwangsweise als Operanden genutzt werden.

Bitte beachten Sie, das falls eine Bedingung zur Ausführung festgelegt wurde, diese vorab zu True (wahr) evaluieren muss.

2.3 BEDINGUNG ZUR AUSFÜHRUNG

Wurde der Auslöser aktiviert, wird im nächsten Schritt geprüft, ob die Bedingung zur Ausführung wahr True (wahr) ist.

Wurde keine Bedingung zur Ausführung festgelegt, so wird die Operation auf jeden Fall ausgeführt. Wurde eine Bedingung zur Ausführung festgelegt, wird die Operation nur ausgeführt, wenn der Vergleich von **zwei Eingangsoperanden** (Konstanten, Kommunikationsobjekte oder interne Variablen) True (wahr) ist.

Die verfügbaren **Vergleichs-Operation** sind wie folgt (abhängig von der Größe der Operation): gleich; ungleich; größer als; größer oder gleich; kleiner und kleiner oder gleich.

2.4 OPERATIONEN

Jede logische Funktion besteht aus der Abfolge von bis zu vier Operationen:

- **Logik**: ID, NOT, AND, OR, XOR, NAND, NOR und NXOR.
- **Arithmetik**: ID, Addition, Subtraktion, Multiplikation, Division, Maximum und Minimum.
- **Vergleich**: gleich; ungleich; größer als; größer oder gleich; kleiner und kleiner oder gleich.
- **Umwandlung**: Löst eine Operation aus, um einen Operand von einem Wert in einen anderen zu wandeln (etwa die Umwandlung eines 1-Bit Wertes in einen 1-Byte Wert).

Das Logikmodul kann die folgende Wertebereiche abdecken (sowohl für Kommunikationsobjekte, interne Variablen mit Zwischenergebnissen, oder numerische Konstanten, die via ETS Parameter festgelegt werden:

- Binärwert: **0** und **1**.
- Ganzzahl o. Vorzeichen (ein-Byte) **0 – 255**.
- Prozentwert (ein-Byte) **0 – 100**.
- Ganzzahl o. Vorzeichen (ein-Byte) **0 – 65535**.
- Ganzzahl (Vorz.) (zwei-Bytes). **-32768 – 32767**.
- Gleitkommazahl Wert (zwei-Bytes) **-671088,64 – 671760,96**.
- Ganzzahl (Vorz.) (vier-Bytes). **-2147483648 – 2147483647**.

Für weitere Informationen bezüglich der Operationen, der Kürzung von Werten bei Umwandlungen, konsultieren Sie bitte *ANNEX I: Referenzen der Operationen*.

2.5 EINGANGSOBJEKTE.

Multiple spezifische Objekte können im Logikmodul aktiviert werden:

- Bis zu 32 ein-Bit Objekte,
- Bis zu 16 ein-Byte Objekte,
- Bis zu 16 zwei-Byte Objekte.
- Bis zu 8 vier-Bit Objekte.

Der Wert dieser Objekte kann beispielsweise als Operand der freigegebenen Funktionen dienen.

2.6 INTERNE VARIABLEN

Zusätzlich stehen die folgenden Variablen zur Verfügung:

- 32 ein-Bit interne Variablen (b1, …, b32),
- 16 ein-Byte interne Variablen (n1, …, n16),
- 16 zwei-Byte interne Variablen (x1, …, x16).
- 8 vier-Byte interne Variablen (y1, …, y8).

Sämtliche dieser Variablen können genutzt werden, um Zwischenergebnisse temporär zu verstauen, die als Eingangswert für weitere Operationen genutzt werden können.

2.7 ERGEBNISOBJEKTE

Jede Logikfunktion korrespondiert mit einem bestimmten Kommunikationsobjekt (ein-Bit, ein-Byte, zwei-Byte oder vier-Byte), durch welches das Endergebnis einer bestimmten internen Variable (festzulegen via Parameter) nach Ablauf der Operationen auf den Bus gesendet wird.

Wahlweise kann dieses Senden jedes Mal bei Ausführung der Operation erfolgen oder zyklisch oder nur, wenn das Ergebnis der Operation von dem Ergebnis der vorherigen Ausführung abweicht.

Es kann ebenfalls festgelegt werden, dass das Ergebnis nur gesendet wird, wenn das Ergebnis in einen bestimmten Wertebereich fällt. Schlussendlich kann auch eine Verzögerung zum Senden des Ergebnisses parametriert werden.

3 ETS PARAMETRIERUNG

3.1 ALLGEMEIN

Die allgemeine Konfiguration der Logikfunktionen enthält die folgenden Optionen: Bitte beachten Sie, dass die Abbildungen zur MAXinBOX 66 gehören, bei anderen Geräten kann es zu geringfügigen Abweichungen kommen.

Wichtig: *Die Parametertabs des Logikmoduls wird eventuell nicht direkt in der ETS angezeigt - in diesem Fall müssen die logischen Funktionen explizit im Allgemeinen Tab des Gerätes freigegeben werden. Bitte schlagen Sie im entsprechenden Produkthandbuch für weitere Details nach.*

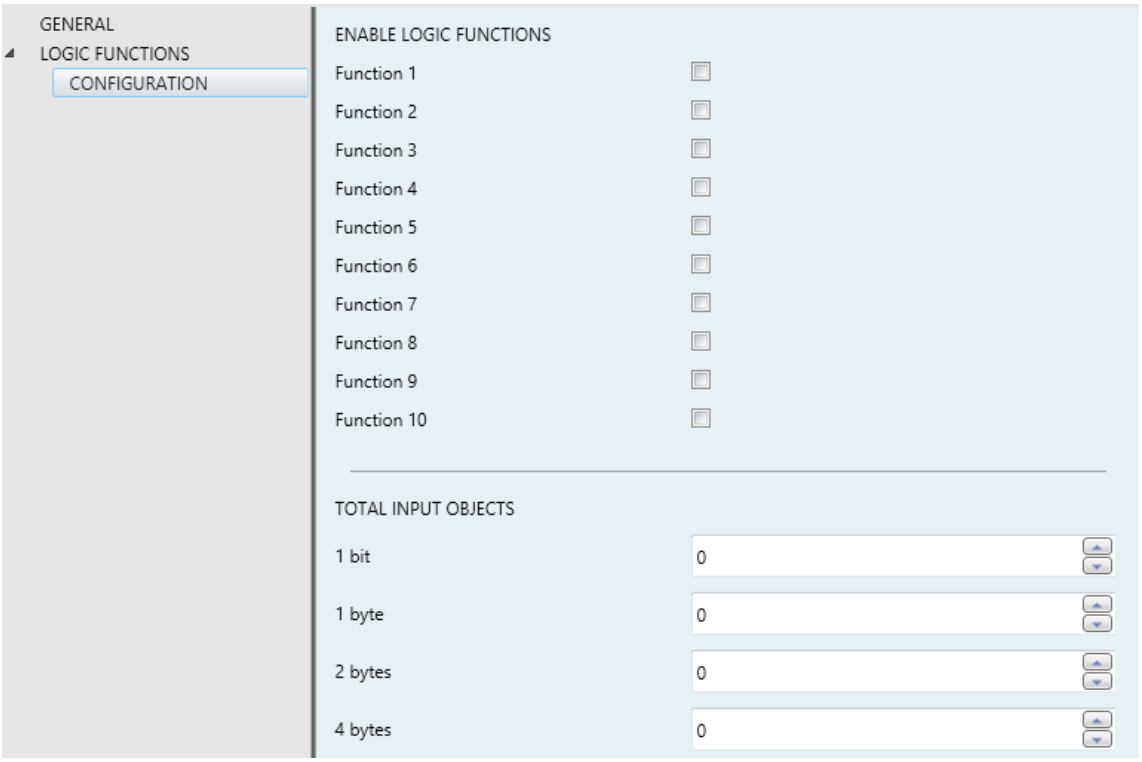


Abb. 2. Allgemeiner Parameter Tab des Logikmoduls.

Wie dargestellt ist keine der 10 Funktionen per default verfügbar. Nach Freigabe der logischen Funktionen, werden die entsprechenden Tabs in der ETS angezeigt.

Des weiteren muss der Integrator zwingend die maximale Anzahl der Eingangsobjekte für alle logischen Funktionen freigeben. Wie bereits in Abschnitt 2.5 dargestellt, ist es möglich, bis zu 32 ein-Bit Objekte, 16 ein-Byte Objekte, 16 zwei-Byte Objekte oder 8 vier-Byte Objekte freizugeben.

Der nächste Abschnitt geht genauer auf die einzelnen Parameter ein.

3.2 FUNKTION n

Pro Funktion wird ein separater Tab (siehe Abschnitt 3.1) in der ETS erscheinen.

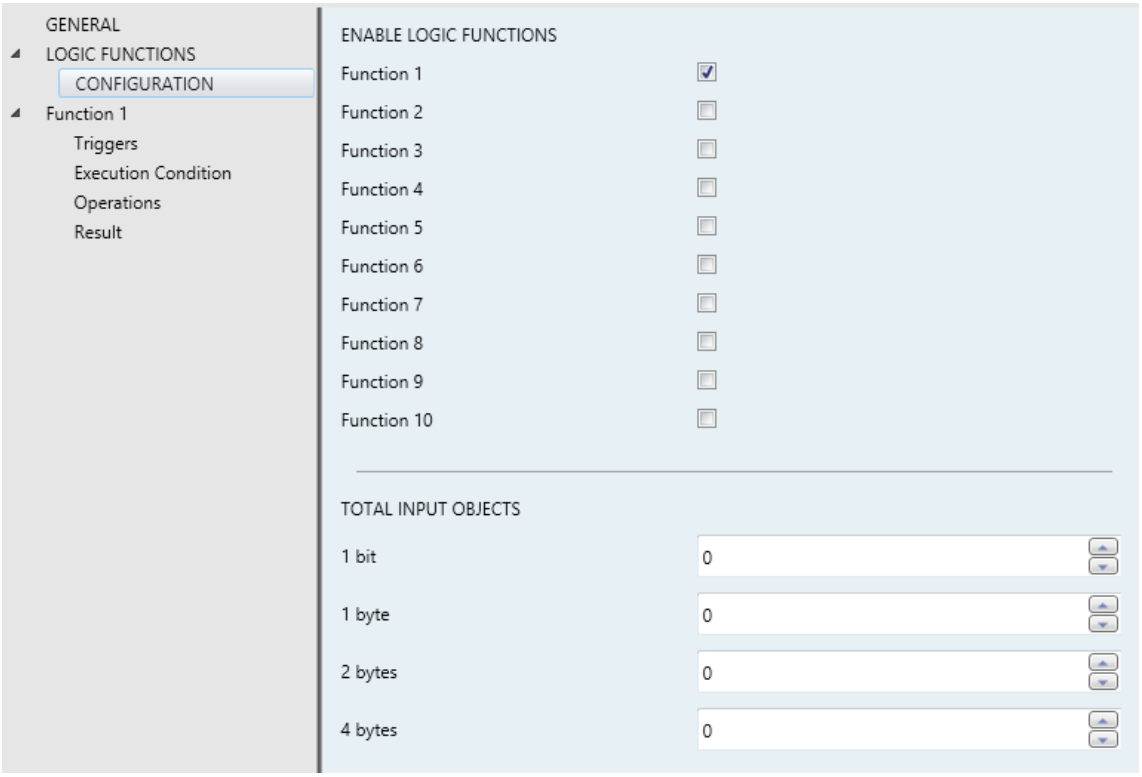


Abb. 3. Funktion 1 aktiviert.

3.2.1 AUSLÖSER

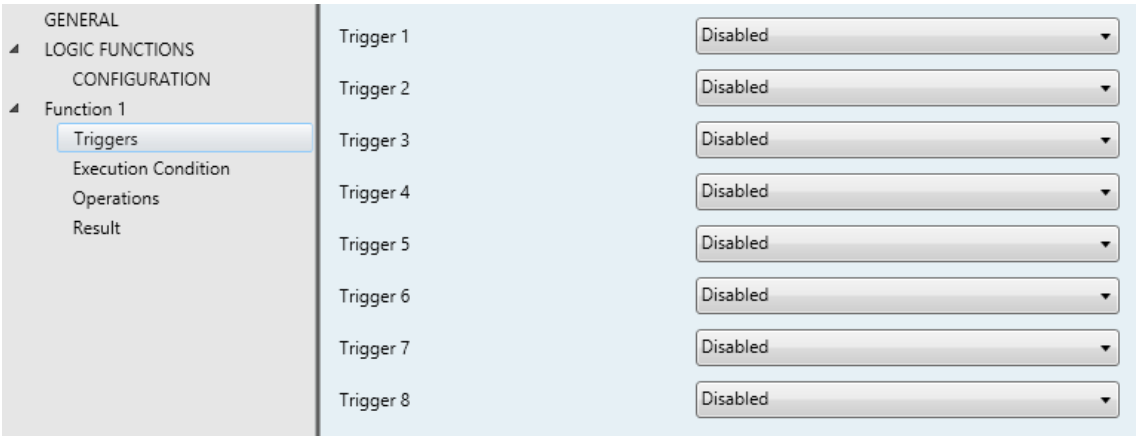


Abb. 4. "Auslöser" Tab

In diesem Abschnitt können bis zu **acht Objekte** als Auslöseobjekte festgelegt werden. Auslöseobjekte lösen bei Empfang des korrespondierenden Wertes und wahrer Bedingung zur Ausführung die entsprechende Funktion aus. Wird nur ein Auslöseobjekt für mehrere Funktionen festgelegt, werde diese Funktionen aufeinander abfolgend ausgeführt, wenn der korrespondierende Wert empfangen wird.

Wichtig: *Auslöseobjekte müssen vorher via ETS freigegeben werden (siehe 3.1)*.

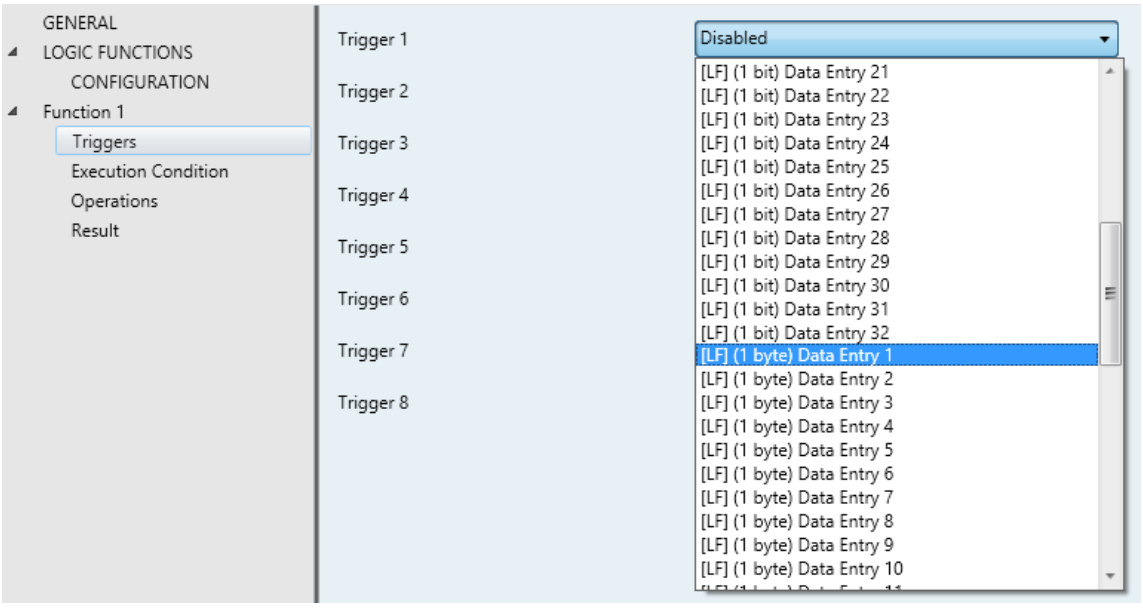


Abb. 5. Festlegen von Auslöseobjekten

3.2.2 BEDINGUNG ZUR AUSFÜHRUNG

In diesem Abschnitt kann die Bedingung zur Ausführung konfiguriert werden:

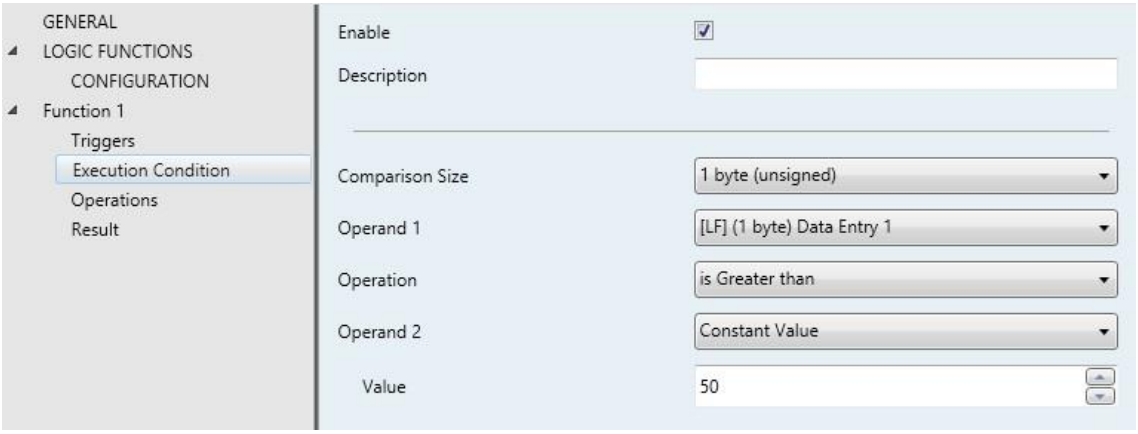


Abb. 6. "Bedingung zur Ausführung" Tab.

- **Aktivieren**: legt fest, ob die Ausführung der Funktion an eine Bedingung geknüpft ist.
- **Beschreibung**: eine kurze Beschreibung (bis zu 100 Zeichen). Hier kann eine kurze Beschreibung der Funktion eingetragen werden.
- **Vergleichsgröße**: legt die Größe der zu vergleichenden Operanden fest: ein-Bit, ein- Byte (ohne Vorz.), ein-Byte (Prozentwert), zwei-Byte (o. Vorz.), zwei-Byte (Vorz.), zwei-Byte (Fließkomma), vier-Byte (Vorzeichen).
- **Operand 1**: Quelle des Wertes: Kommunikationsobjekt, interne Variable oder Konstante. Wird "Konstante" ausgewählt, wird "**Wert**" angezeigt, um die gewünschte Konstante einzutragen.
- **Operation**: legt die Größe des Vergleichs fest. Wird 1-Bit gewählt, zeigt dieses Feld die Optionen "Gleich" und "Ungleich". Wird die Vergleichsgröße auf einen anderen Wert gesetzt, so wird die komplette Liste an Optionen angezeigt (siehe Abschnitt 2.4).
- **Operand 2**: Quelle des Wertes: Kommunikationsobjekt, interne Variable oder Konstante. Wird "Konstante" ausgewählt, wird "**Wert**" angezeigt, um die gewünschte Konstante einzutragen.

3.2.3 OPERATIONEN

Die maximal mögliche Anzahl an Operationen pro Logikfunktion beträgt vier. Jede wird individuell aktiviert und nacheinander ausgeführt. Wird eine der Operationen deaktiviert, wird diese bei Ausführung ignoriert.

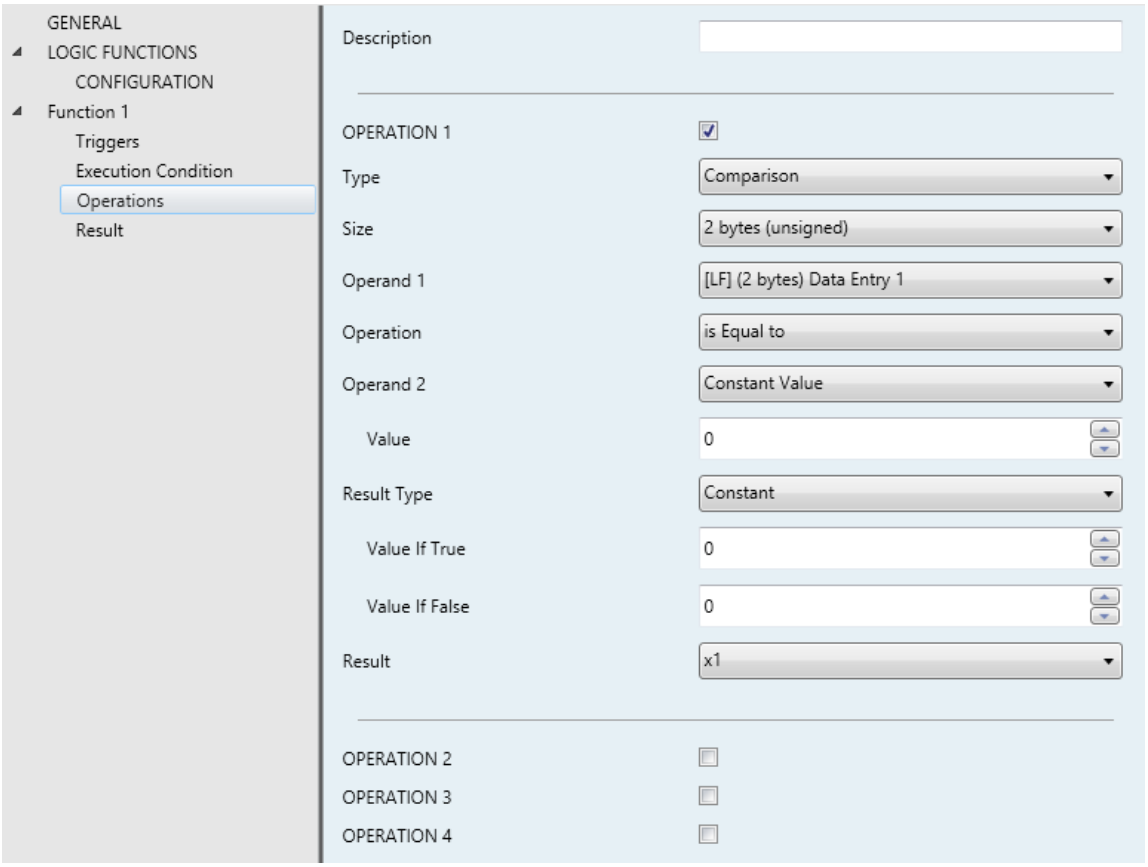


Abb. 7. "Operationen" Tab

Hierfür können die folgenden Parameter konfiguriert werden:

- **Beschreibung**: eine kurze Beschreibung (bis zu 100 Zeichen). Hier kann eine kurze Beschreibung der Funktion eingetragen werden.

- **Operation "i"**: aktiviert oder deaktiviert die entsprechende Operation "i" (1-4). Jede aktivierte Operation bietet die folgenden Parameter:

  - **Typ**: legt den Typ der Operation fest (Logik, Arithmetik, Vergleich oder Umwandlung). Wird eine andere Auswahl als "Logik" gewählt, so wird ein weiterer Parameter ("**Größe**") freigegeben für die Auswahl der Größe des Ergebnisses der Operation ("ein-Bit", "ein-Byte

(O. Vorz.)", "ein-Byte (Prozentwert)", "zwei-Byte (o. Vorz.)", "zwei-Byte (Vorz.)", "zwei-Byte (Fließkomma)" und "vier-Byte (Vorz.)"). Siehe Abschnitt 2.4.

- **Operation**: legt die Aktion fest, die ausgeführt wird in Operation Nummer "i". Abhängig von der Art der Logikoperation (Logik, Arithmetik, Vergleich oder Umwandlung), bietet dieser Parameter unterschiedliche Optionen. Für weitere Informationen bitte im folgenden Abschnitt nachschlagen *ANNEX I: Referenzen der Operationen*.

- **Operand "j"**: abhängig von der gewählten Operation werden mehrere neue Parameter freigegeben ("Operand j") mit denen die Eingangswerte (Operanden) festgelegt werden können. Dies können sowohl Kommunikationsobjekte, interne Variablen oder Konstantwerte sein (siehe Abschnitt 2.4 und weiter).

- **Ergebnis**: bestimmt die interne Variable, in der das Ergebnis der Operation gespeichert wird. Dieses Teilergebnis kann später als Endergebnis der Funktion verwendet werden oder in weiterer Operationen als Operand dienen.

Wichtig: *Alle logischen Funktionen teilen das Set an internen Variablen miteinander. So ist es beispielsweise so, dass Funktion 1 etwa ein Teilergebnis in Variable "n1" speichert und anschließend Funktion 2 diese Variable mit dem von Funktion 1 gespeicherten Wert verwendet.*

3.2.4 ERGEBNIS

In diesem Abschnitt ist es möglich, festzulegen, welche interne Variable vorgesehen ist, um das Endergebnis der Funktion abzuspeichern, um nach Ablauf aller Operationen den entsprechenden Wert über das Kommunikationsobjekt "**[FL] Funktion *n* – Ergebnis**" auf den Bus zu senden.

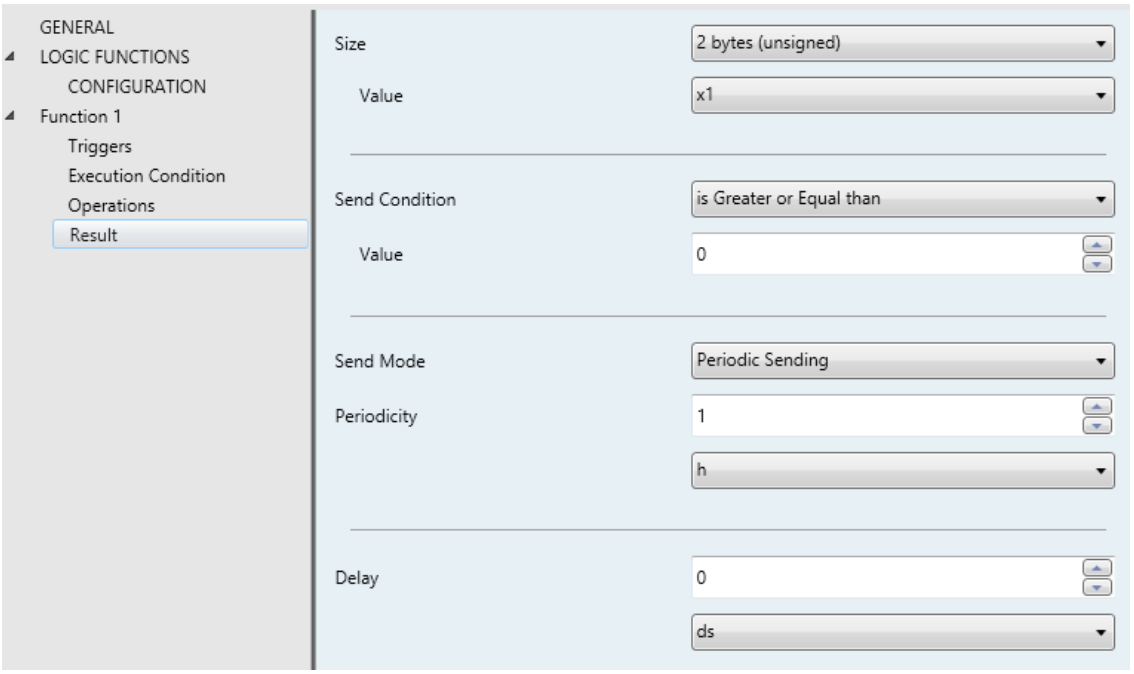


Abb. 8. "Ergebnis" Tab.

- **Größe**: legt die Größe des Funktionsergebnisses fest. Die möglichen Optionen sind: „ein-Bit, ein-Byte (ohne Vorz.), ein-Byte (Prozentwert), zwei-Byte (o. Vorz.), zwei-Byte (Vorz.), zwei-Byte (Fließkomma), vier-Byte (Vorzeichen).

- **Wert**: legt die interne Variable fest, deren Wert durch das Ergebnisobjekt nach Ablauf aller Operationen auf den Bus gesendet wird.

- **Sendebedingung**: legt die Bedingung zum Senden fest, so dass nur die gewünschten Werte gesendet werden. Die möglichen Optionen sind:

  - **Für ein-Bit Ergebnis**:

    - Keine Bedingung,
    - gleich,
    - ungleich.

- **Für Ergebnisse anderer Größen (1 byte, 2 bytes, 4 bytes)**:

  - Keine Bedingung,
  - gleich,
  - ungleich.
  - größer,
  - Größer oder gleich,
  - kleiner,
  - kleiner oder gleich.

  Wurde eine Bedingung gewählt, muss die entsprechende im Parameter "**Wert**" hinterlegt werden. Der verfügbare Wertebereich orientiert sich an der gewählten Größe (siehe Abschnitt 2.4).

- **Sendemodus**: hier werden die Umstände festgelegt, unter denen das Ergebnis der Funktion gesendet wird, sollte die Sendebedingung erfüllt sein.

  - Immer.

  - Bei Änderung des Endergebnisses: das Ergebnis der Funktion wird nur gesendet, wenn der aktuelle Wert sich von vorherigen unterscheidet.

  - Zyklisches Senden: das Ergebnisobjekt wird regelmäßig auf den Bus gesendet, sobald die Funktion ausgelöst wurde. "**Periodizität**" legt den gewünschten Zyklus fest. Der Wertebereich reicht von 10 bis 600 Zehntelsekunden; 1 bis 3600 Sekunden; 1 bis 1440 Minuten und 1 bis 24 Stunden.

    Bitte beachten Sie, dass in diesem Fall, dass **das Ergebnis erneut evaluiert wurde**, um das Ergebnis zu aktualisieren muss die Funktion erneut ausgelöst werden. Ein erneutes Auslösen der Funktion unterbricht das zyklische Senden und setzt den Zyklustimer zurück, auch wenn das **neue Ergebnis nicht auf den Bus gesendet wird, da es die Bedingung nicht erfüllt**. In diesem Fall wird auch das vorherige Ergebnis nicht weiter gesendet.

- **Verzögerung**: legt den Zeitraum zwischen Ausführung der Funktion und Senden des Ergebnisses fest. Soll das Ergebnis unverzüglich gesendet werden, so muss der Wert "0" hinterlegt werden.

Der mögliche Wertebereich hängt von der gewählten Zeiteinheit ab: Der Wertebereich reicht von 0 bis 600 Zehntelsekunden; 1 bis 3600 Sekunden; 1 bis 1440 Minuten und 1 bis 24 Stunden.

Die Verzögerung bezieht sich nur auf das Senden des Ergebnisses und ist unabhängig vom Auslösen der Funktion. Aus diesem Grund wird das Ergebnis während dem Verzögerungstimer nicht durch Änderung der Operanden beeinflusst.

Wichtig: *Wenn eine Funktion erneut durch ein vorheriges Ergebnis ausgelöst wird und dieses zum Senden an den Bus (aufgrund der Verzögerung) ansteht, wird die Funktion erneut ausgeführt und überschreibt somit das vorherige Ergebnis und startet die Zeitzählung der Verzögerung erneut.* Falls das neue Ergebnis ***die Bedingung zum Senden nicht erfüllt*** *wird die Verzögerung abgebrochen*.

ANNEX I: REFERENZEN DER OPERATIONEN

BOOL´SCHE LOGIK (1 BIT)

- **ID (identity)**

1. Operand	*2. Operand*	*Ergebnis*
0	-	0
1	-	1

- **AND (UND-Verknüpfung)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	0
0	1	0
1	0	0
1	1	1

- **OR (ODER-Verknüpfung)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	0
0	1	1
1	0	1
1	1	1

- **XOR (exklusive OR)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	0
0	1	1
1	0	1
1	1	0

- **NOT (Negation)**

1. Operand	*2. Operand*	*Ergebnis*
0	-	1
1	-	0

- **NAND (negierte UND)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	1
0	1	1
1	0	1
1	1	0

- **NOR (negierte ODER)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	1
0	1	0
1	0	0
1	1	0

- **NXOR (negierte XOR)**

1. Operand	*2. Operand*	*Ergebnis*
0	0	1
0	1	0
1	0	0
1	1	1

ARITHMETISCHE OPERATIONEN

- **ID (identity)**

1. Operand	*2. Operand*	*Ergebnis*
v1	-	*v1*

- **ADDITION**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*v1 + v2*

- **SUBTRAKTION**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*v1 - v2*

- **MULTIPLIKATION**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*v1 * v2*

- **DIVISION**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*v1 / v2*

- **Maximum**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*max (v1, v2)*

- **MINIMUM**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*min (v1, v2)*

Wichtig: *Es ist ratsam, die Zusätzlichen Bemerkungen zu lesen für weitere Informationen bezüglich Sonderfällen und Bereichsüberschreitungen*.

VERGLEICHE

- **GRÖßER**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*1↔ v1 + v2. 0 in jedem anderen*

- **GRÖßER ODER GLEICH**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*1 ↔ v1 ≥ v2. 0 in jedem anderen*

- **KLEINER**

1. Operand	*2. Operand*	*Ergebnis*

v1	*v2*	*1 ↔ v1 < v2.* *0 in jedem anderen*

- **KLEINER ODER GLEICH**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*1 ↔ v1 ≤ v2.* *0 in jedem anderen*

- **GLEICH**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*1 ↔ v1 = v2.* *0 in jedem anderen*

- **UNGLEICH**

1. Operand	*2. Operand*	*Ergebnis*
v1	*v2*	*1 ↔ v1 ≠ v2.* *0 in jedem anderen*

Wichtig: *Es ist ratsam, die Zusätzlichen Bemerkungen zu lesen für weitere Informationen bezüglich Sonderfällen und Bereichsüberschreitungen*.

UMWANDLUNGS-OPERATIONEN

1-Bit Operand

	1byte (ohne Vorzeichen)	*2byte (ohne Vorzeichen)*	*2byte (Vorzeichen)*	*2byte (Fließkomma)*	*4Byte (Vorzeichen)*
0	*0*	*0*	*0*	*0.00*	*0*
1	*1*	*1*	*1*	*1.00*	*1*

Ein-Byte Operand ohne Vorzeichen

	1 bit	*2 bytes (Ohne Vorzeichen)*	*2 bytes (Vorzeichen)*	*2 bytes (Fliesskomma)*	*4 bytes (Vorzeichen)*
0	*0*	*0*	*0*	*0.00*	*0*
10	*1*	*10*	*10*	*10.00*	*10*
180	*1*	*180*	*180*	*180.00*	*180*
255	*1*	*255*	*255*	*255.00*	*255*

Ein-Byte Prozentwert Operand

	1 bit	1 byte (Ohne Vorzeichen)	2 bytes (Ohne Vorzeichen)	2 bytes (Vorzeichen)	2 bytes (Fliesskomma)	4 bytes (Vorzeichen)
0	0	0	0	0	0.00	0
10	1	10	10	10	10.00	10
80	1	80	80	80	80.00	80
100	1	100	100	100	100,00	100

Zwei-Byte Operand ohne Vorzeichen

	1 bit	1 byte (Ohne Vorzeichen)	2 bytes (Vorzeichen)	2 bytes (Fliesskomma)	4 bytes (Vorzeichen)
0	0	0	0	0.00	0
10	1	10	10	10.00	10
500	1	255	500	500.00	500
65535	1	255	32767	65535.00	65535

Zwei-Byte Operand mit Vorzeichen

	1 bit	1 byte (Ohne Vorzeichen)	2 bytes (o.Vorz.)	2 bytes (Fliesskomma)	4 bytes (Vorzeichen)
-32768	0	0	0	-32768.00	-32768
-12000	0	0	0	-12000.00	-12000
0	0	0	0	0.00	0
12345	1	255	12345	12345.00	12345

zwei-Byte Fliesskomma Operand

	1 bit	1 byte (Ohne Vorzeichen)	2 bytes (o.Vorz.)	2 bytes (Vorzeichen)	4 bytes (Vorzeichen)
-671088,64	0	0	0	-32768	-671088
-321654,98	0	0	0	-32768	-321654
0,00	0	0	0	0	0
12345,67	1	255	12345	12345	12345

Vier-Byte Operand mit Vorzeichen

	1 bit	1 byte (Ohne Vorzeichen)	2 bytes (o.Vorz.)	2 bytes (Vorzeichen)	2 bytes (Fliesskomma)
-2147483648	0	0	0	-32768	0xF800 (*)
-1	0	0	0	-1	-1,00
0	0	0	0	0	0,00
123456	1	255	65535	32767	123456,00
2147483647	1	255	65535	32767	0x7FFF (*)

(*) Die Minimum- und Maximumwerte bei 2-Byte Fliesskommazahlen arbeiten nach KNX Standard als –unendlich und +unendlich, wie beschrieben in Zusätzliche Bemerkungen.

ZUSÄTZLICHE BEMERKUNGEN

Wie bereits beschrieben arbeiten Logikmodule mit den folgenden Datentypen:

- Binär: **0** und **1**.
- Ganzzahlen ohne Vorzeichen
  - Ein Byte **0 – 255**.
  - Zwei Byte: **0– 65535**
- Prozentwert (ein-Byte) **0 – 100**.
- Ganzzahlen mit Vorzeichen
  - Zwei Byte: **-32768 – 32767**.
  - Vier Byte: **-2147483648 – 2147483647**.
- Gleitkommazahl (zwei-Bytes) **-671088,64 – 671760,96**.

Diese Operanden können sowohl Kommunikationsobjekte, interne Variablen für Teilergebnisse oder sogar numerische Konstanten, die via Parameter für bestimmte Operationen festgelegt werden, sein.

Bitte beachten Sie die folgenden Besonderheiten:

- Bei **Bereichsüberschreitung bei arithmetischen Operationen** wird der überschüssige Wert zurückgemeldet. So wird bei einer ein-Byte Addition von 255 und 10 der Wert 255 der als obere Schwelle für ein-Byte Werte erlaubt ist, zurückgemeldet.
- **Spannungsausfälle** des KNX Bus führen nicht zu Datenverlust, die Variablen und Objekte behalten die entsprechenden Werte.
- **Divisionen durch Null** ergeben kein Resultat, eine Funktion mit einer „Teilen durch Null“ Operation wird abgebrochen.
- **Multiplikation und Division von Prozentwerten miteinander** werden nach dem folgenden Beispiel behandelt:

- Zwei Konstanten:

  Bei Multiplikation oder Division zweier Prozentwerte wird eine der beiden als Ganzzahl angesehen. Wenn bei einer Multiplikation Operand No. 1 auf “25” und Operand No. 2 auf “2” gesetzt wird, ist das Ergebnis “50%” (Bitte beachten Sie, dass das Ergebnis vorher bekannt ist).

- Eine Konstante und ein Objekt:

  Diese Option ist etwa für Fälle, in denen ein Jalousiemotor „k“ Male (wobei „k“ eine Konstante ist) die Position einer Jalousie fahren soll (welche via Kommunikationsobjekt gesendet wird). Wenn "k” auf den Wert “3” gesetzt wird und die aktuelle Position der Jalousie “30%” beträgt, dann wird der Jalousiemotor auf 90% fahren.

- Zwei Objekte:

  - Wenn, wie oben angesprochen, eine Jalousie “B” “k”-male die Position einer Jalousie “A” anfahren soll, “k” in diesem Fall jedoch eine Ganzzahl ohne Vorzeichen (ein Byte) sein soll (keine Konstante), so muss “k” zuerst in eine interne Variable konvertiert werden, um anschließend mit dem Statusobjekt von B multipliziert werden zu können.

  - Wenn im gleichen Beispiel “k%” ein ein-Byte Prozentwert ist (und keine Konstante): Um B “k%“ der Position von A fahren zu lassen (etwa 50% von 40%, also 20%) ist es notwendig, beide Werte in zwei-Byte Fliesskommawerte zu wandeln, diese zu multiplizieren und das Ergebnis in einen ein-Byte Prozentwert zu konvertieren.

- Der obere (positivste) und niedrigste (negativste) Wert von Fliesskommazahlen im KNX Standard sind fest definiert (i.e., -671088.64 angezeigt als 0xF800, und +670760.96 angezeigt als 0x7FFF) und operieren als **+∞ (plus unendlich) und –∞ (minus unendlich)**:

  - (670760.96) * (-671088.64) = (-671088.64)
  - (-671088.64) * (-671088.64) = (670760.96)
  - (670760.96) * (34.00) = (670760.96)
  - (670760.96) * (-34.8) = (-671088.64)
  - (670760.96) / (65.2) = (670760.96)

- (670760.96) / (-341.12) = (-671088.64)
- (120000.00) / (670760.96) = (0.00)
- (-671088.64) / (670760.96) = (-1.00)
- (-671088.64) / (3500.66) = (-671088.64)

- Wenn sowohl ein **wiederholtes Senden** und eine **Verzögerungszeit** parametriert wurden, wird die Verzögerungszeit nur VOR dem ersten Senden angewendet. Nachdem die Verzögerungszeit abgelaufen ist, wird das Ergebnis auf den Bus gesendet und der Timer für das erneute Senden startet. Die Verzögerung bezieht sich NICHT auf das wiederholte Senden, sondern wird nur beim erneuten Auslösen der Funktion angewendet.

Besuchen Sie uns und senden Sie uns Ihre Anregungen über Zennio Produkte:
http://support.zennio.com

Zennio Avance y Tecnología S.L.
C/ Río Jarama, 132. Nave P-8.11
45007 Toledo (Spain).

Tel. +34 925 232 002.

www.zennio.com
*info@*zennio*.com*