BEDIENUNGSANLEITUNG

SKX Open

SKX ADVANCE

ZN1RX – RS232

Edition 1

Version 1.1

INHALT

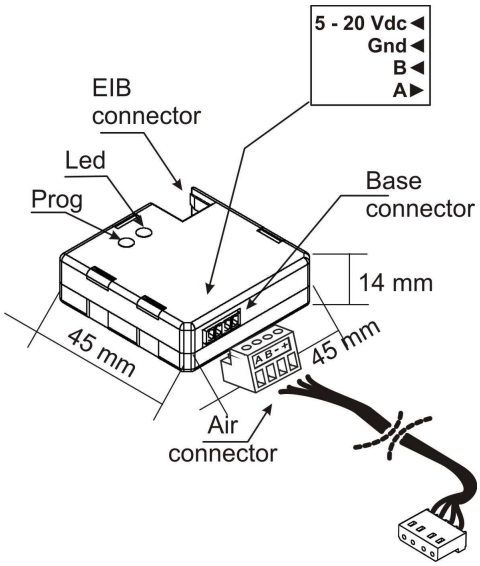
1. EINLEITUNG

1.1. ZN1RX-SKX OPEN

Der ZN1RX-SKX OPEN ist das Zennio-Produkt welches den Datenaustausch zwischen einem KNX-Bus und einem RS 232 Datenbus ermöglicht.

Geräteeigenschaften

- Reduzierte Abmessungen: 45 x 45 x 14mm.
- Geeignet zum Einbau in Abzweigdosen, Gerätedosen, oder in Schaltschränken.
- Mehrere Geschwindigkeiten und Fehlerkorrektionsmechanismen.
- Ideal für M2M-Applikationen
- Basiert auf einem BIMM112-Modul
- Kompletter Datenerhalt.
- Erfüllt CE Standard.



Beschreibung der Elemente

- **Prog:** Durch das Betätigen der Programmiertaste wird das Gerät in den Programmiermodus gebracht. Bei Erstbetätigung, nach Anlegen der Busspannung, geht das Gerät in den "Sicherheitsmodus".
- **LED:** Lichtsignal, zeigt an dass sich das Gerät im Programmiermodus befindet. Befindet sich das Gerät im Sicherheitsmodus, blinkt sie in einem Intervall von 0,5 Sek.

1.2. APPLIKATIONSPROGRAMM: SKX ADVANCE

Der Zweck dieser Bedienungsanleitung ist es das speziell entwickelte Applikationsprogramm zur Integration von Geräten die über eine RS 232 Schnittstelle gesteuert werden können, zu erklären.

Der SKX Advance ist ein Produkt, welches zur Kommunikation zwischen einem KNX-System und einem RS 232 Protokoll dient. Unabhängig vom verwendeten Hexadezimalcode den ein Gerät mit einer Aktion asoziiert, kann dieser mit einem Kommunikationsobjekt verknüpft werden um Befehle von KNX-Geräten zu empfangen oder an diese zu senden.

Bitte beachten: Es können beliebige Hexadezimalcodes integriert werden, so lange sie sich innerhalb der vom SKX Open bestimmten Einschränkungen in Bezug auf die Länge dieser Codes befinden.. (Siehe Abschnitt *3.2 Kommunikationskapazität*)

Die Kommunikation wird über den SKX Open ausgeführt, dadurch wird ein bidirektionaler Datenaustausch ermöglicht. Das heisst es können Daten von KNX-Geräten zum angeschlossenen Datenbus, und umgekehrt von diesem an die KNX-geräte gesendet werden.

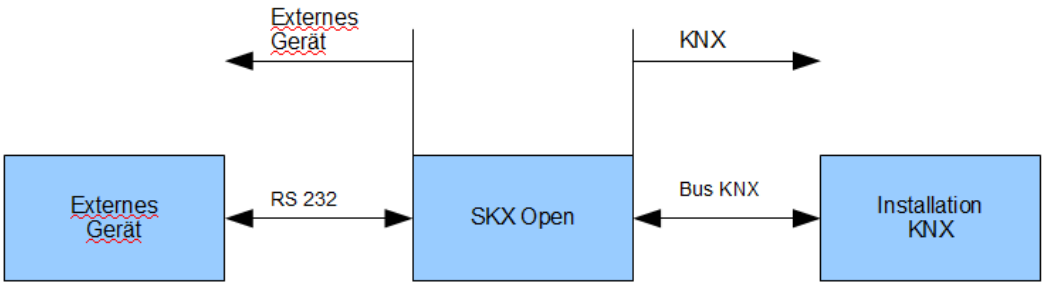


Bild 1. Kommunikation SKX Open

1.3. GRUNDLEGENDE SPEZIFIKATIONEN DES SKX ADVANCE

In diesem Abschnit werden die grundlegenden Spezifikationen des SKX Advance beschrieben:

- Übertragungsgeschwindigkeit (1200, 2400, 4800, 9600, 19200 Baud)
- Parität: Ohne, Gerade oder Ungerade
- Erkennung Übertragungsende der Frames (Timeout, End-Byte)
- Anzahl der Kommunikationsobjekte: 65

Grösse der Objekte	Anzahl der Objekte
1 bit	40
1 byte	20
14 byte	5

- Fehlererkennung: 1-bit Fehlerobjekte
- Maximale Länge des Protokolls: 29 byte / 58 HEX-Zeichen (es können bis zu 10 feste byte bzw. 20 feste HEX-Zeichen pro Frame definiert werden) einschliesslich Sonderzeichen welche eine Einfügung von Kopfzeilen, Subframes, Checksumme etc. ermöglicht.

WARNUNG

Es sollte mit besonderer Vorsicht bei der Integration eines beliebigen Gerätes in ein KNX-System vorgegangen werden, da dadurch eine Möglichkeit der externen Manipulation des jeweils anderen Systems geschaffen wird.

2. INSTALLATION

2.1. INSTALLATION DES SKX OPEN AN DAS KNX-BUSSYSTEM

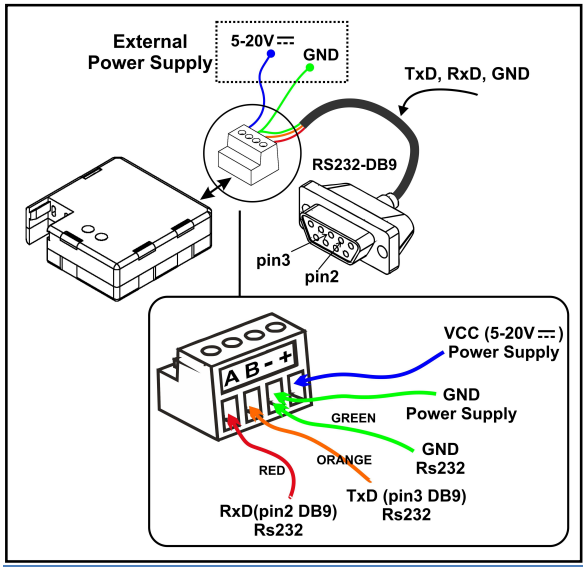
Der SKX Open wird wie jedes beliebige andere KNX-Gerät an den Bus angeschlossen. Hierzu muss das Gerät nur mit Hilfe einer Standardbusklemme mit dem KNX-Bus verbunden werden, und kann dann programmiert werden.

Ist das Gerät einmal mit der Busspannung versorgt, so kann die physikalische Adresse vergeben werden, sowie das Applikationsprogramm SKX Advance aufgespielt werden.

Dieses Gerät benötigt keine externe Spannungsversorgung, es funktioniert ausschliesslich mit der KNX-Busspannung. Es ist jedoch notwendig den RS 232 Datenbus mit einer eigenen, dem Standard entsprechenden Spannung zu versorgen.

2.2. ANSCHLUSS DES SKX OPEN AN DIE RS 232 SCHNITTSTELLE

Die Verbindung mit dem RS 232 Datenbus wird über eine spezifische Steckklemme realisiert, wodurch die Installation erleichtert wird. Nachfolgend wird die Verbindung der beiden Systeme beschrieben.



Klemme des SKX Open	BUS RS232
A	RSA
B	RSB
-	Masse
+	+12V

Bild 2. Anschluss des SKX Open

3. PARAMETRISIERUNG

Das für das Applikationsprogramm SKX Advance verwendete Gerät, ist das existente Zennio-Produkt ZN1RX-RS232, oder SKX Open. Dank dieses Applikationsprogramms ist es möglich jedes beliebige Gerät mit RS232-Schnittstelle zu integrieren, unter der Vorraussetzung dass die mit jeder Aktion verknüpften Hexadezimalcodes bekannt sind.

Der SKX Advance verfügt über 65 Kommunikationobjekte unterschiedlicher Grösse über die KNX und RS232 miteinander kommunizieren. Ausserdem stehen mehrere 1-bit Objekte zur Verfügung, mit denen Fehler die während der Applikation auftreten können identifiziert werden (Ungerade Frame-Länge, falsche Benutzung von '#', zu lang, Empfangsfehler am RS232-Port, Einsatz von nicht-hexadezimalen Zeichen)

Abhängig von ihrer Grösse der Objekte existieren bis zu elf verschiedene Parametermöglichkeiten diese zu steuern:

- **1-bit Objekte:** ermöglichen das Senden eines festen (definierten) Frames an den RS232-Port wenn auf diesem Objekt ein Wert vom Bus empfangen wird, oder das Senden eines 1-bit Werts auf den Bus bei Empfang eines festen Frames vom RS232-Port.

Komunikation KNX => RS232. 1-bit Befehle.

- **Senden eines Frames** (als Parameter eingegeben) an das über den seriellen Port integrierte Gerät **bei Empfang einer 1** auf dem Kommunikationsobjekt.

Beispiel:

- **Senden eines Frames** (als Parameter eingegeben) an das über den seriellen Port integrierte Gerät **bei Empfang einer 0** auf dem Kommunikationsobjekt.

Beispiel:

Komunikation RS232 => KNX . 1-bit Befehle.

- **Senden einer 0** über das Kommunikationsobjekt bei **Empfang eines Frames über den seriellen Port,** der mit dem als Parameter angegebenen Frame übereinstimmt.

Beispiel:

- **Senden einer 1** über das Kommunikationsobjekt bei **Empfang eines Frames über den seriellen Port,** der mit dem als Parameter angegebenen Frame übereinstimmt.

Beispiel:

- **1-byte Objekte:** ermöglichen das Senden eines Frames an den RS232-Port wenn auf diesem Objekt ein Wert vom Bus empfangen wird, oder das Senden eines Werts auf den Bus bei Empfang eines Frames vom RS232-Port. Die Frames können fest oder variabel, abhängig vom Wert des 1-byte objekts sein. Das gleiche gilt für den Objektwert, der abhängig vom empfangenen Frame oder fest sein kann.

Komunikation RS232 => KNX . 1-bit Befehle.

- **Senden eines festen Frames** (als Parameter eingegeben) an das über den seriellen Port integrierte Gerät **bei Empfang eines definierten Werts** auf dem Kommunikationsobjekt.

Beispiel:

- **Senden eines variablen Frames** (als Parameter eingegeben) an das über den seriellen Port integrierte Gerät **abhängig vom erhaltenen Wert (mittels Schlüssel ##)** auf dem Kommunikationsobjekt. Der gesendete Frame ist der vom Benutzer definierte, wobei die Zeichen ## durch den 1-byte Wert des Kommunikationsobjekts ersetzt werden.

Beispiel:

- **Senden eines variablen Frames (Prozentmodus)** (als Parameter eingegeben) an das über den seriellen Port integrierte Gerät **abhängig vom erhaltenen Wert (mittels Schlüssel ##)** auf dem Kommunikationsobjekt. Der empfangene Wert wird in einen Wert 0-100 umgeformt und in den vom Benutzer definierten Frame eingefügt. Der gesendete Frame ist der vom Benutzer definierte, wobei die Zeichen ## durch den 1-byte Wert des Kommunikationsobjekts ersetzt werden (als Prozentwert)

Beispiel:

Komunikation RS232 => KNX . 1-bit Befehle.

- **Empfangen eines festen Werts** (als Parameter eingegeben) über das Kommunikationsobjekt bei **Empfang eines Frames über den seriellen Port,** der mit dem als Parameter angegebenen Frame übereinstimmt.

Beispiel:

- **Empfangen eines variablen Werts** über das Kommunikationsobjekt **bei Empfang eines Frames über den seriellen Port,** der mit dem als Parameter angegebenen Frame übereinstimmt (mittels Schlüssel ##). Der Wert des Kommunikationsobjekt ist der dem ## -Teil des Frames entsprechendem Wert.

Beispiel:

- **Empfangen eines variablen Werts (Prozentwert)** über das Kommunikationsobjekt **bei Empfang eines Frames über den seriellen Port,** der mit dem als Parameter angegebenen Frame übereinstimmt (mittels Schlüssel ##). Der Wert des Kommunikationsobjekt ist der dem ## -Teil des Frames entsprechendem Wert (Prozentwert).

Beispiel:

- **14-byte Objekte**: ermöglichen das Senden oder Erkennen von Text-Strings innerhalb von seriellen Frames. Notwendigerweise muss angegeben werden wie das Ende eines Strings erkannt werden soll, über *Identifikation String-Ende* oder über *Feste Zeichenanzahl.*

- **Übertragung** eines vom seriellen Port empfangenen variablen Text-Strings auf den KNX-Bus, und umgekehrt..

Beispiel:

3.1. ALLGEMEINES KONFIGURATIONSFENSTER

Dieses Fenster dient zur Konfiguration verschiedener Parameter die die Kommunikation als solches betreffen.

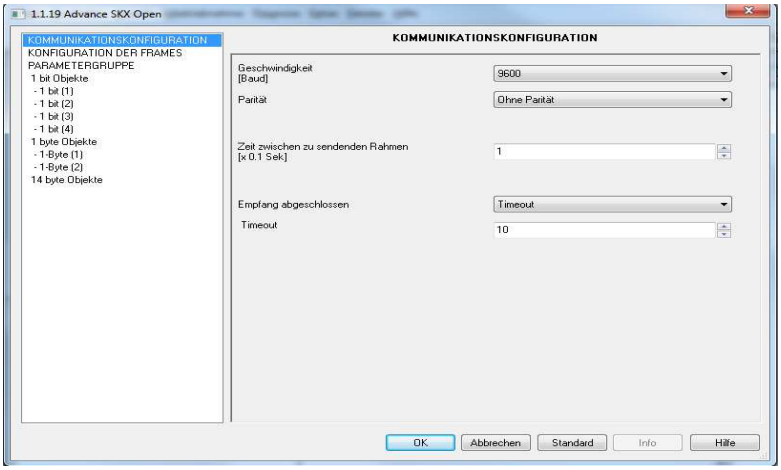


Bild 3. SKX Advance – Konfiguration der Kommunikation

- **Geschwindigkeit:** 1200-2400-4800-9600-19200 Baud
- **Parität:** Ohne Parität, Gerade oder Ungerade
- **Zeit zwischen zu sendenden Frames (in Zehntelsekunden):** Hierbei handelt es sich um eine konfigurierbare Sendepause zwischen den einzelnen Frames. Dieser Parameter ermöglicht dem SKX Advance, für den Fall dass mehrere Kommunikationsobjekte mit einer Gruppenadresse verknüpft sind, die den Objekten zugeordneten Frames so zu senden, dass sie vom Empfänger perfekt verarbeitet und interpretiert werden können. Dieser Parameter ist abhängig von den Charakteristiken des Empfängers.

- **Empfang abgeschlossen:** Zur Erkennung des Frame-Endes stehen zwei Möglichkeiten zur Verfügung:

  - **Timeout:** Dies ist die Zeit nach deren Ablauf ohne Empfang eines bits der Frame als gültig angesehen wird.

  - **End-Byte Datenrahmen:** es besteht die Möglichkeit das Ende eines Frames mit einem eindeutigen,spezifischen Byte zu kennzeichnen. Diese Option stellt zusätzlich ein Timeout als Sicherheit zur Vermeidung von Kommunikationsfehlern zur Verfügung. Im Falle des Empfanges eines Frames mit einer Länge von mehr als 10 byte, werden diese Daten ignoriert und über ein Kommunikationsobjekt ein Fehlertelegramm auf den Bus gesendet.

  -

__Beispiel:__ Ein externes Gerät benötigt 80ms zum Senden des kompletten Frames.

***Erster Fall**: Der Benutzer definiert ein "Timeout" von 30 ms Es wird angenommen dass das externe Gerät einen zweiten Frame direkt nach dem ersten senden will. Diese Funktionsweise wird im folgenden Diagramm dargestellt:*

Nach Ende des ersten Frames beginnt der Timeout, der nächste Frame startet bevor der Timeout abgeschlossen ist, d.h. der Timeout wird abgebrochen und zählt wieder von vorne nach Ende des zweiten Frames. Da der Timeout in diesem Fall abgeschlossen wurde (30 ms), betrachtet der SKX Advance den Frame als beendet. Da allerdings zwei Frames gesendet wurden bevor der Empfang als abgeschlossen betrachtet wurde, interpretiert der SKX Advance den Frame als unbekannt und sendet nichts.

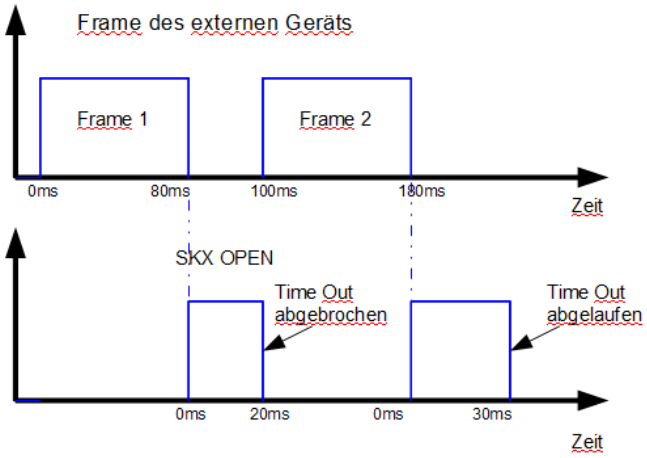


Bild 4. Timeout zu lang

***Zweiter Fall:** Der Benutzer definiert ein "Timeout" von 10 ms Diese Funktionsweise wird im folgenden Diagramm dargestellt:*

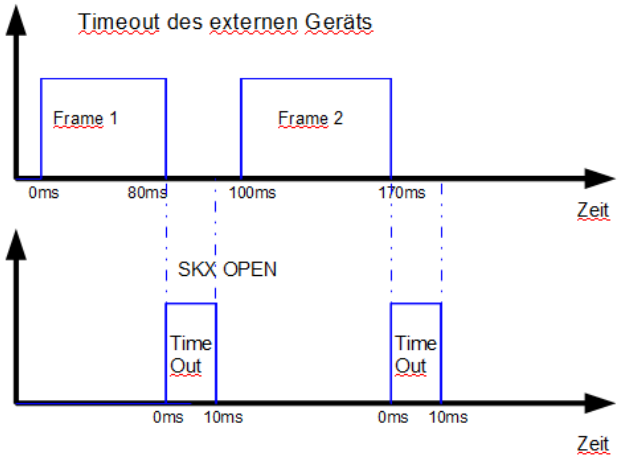


Bild 5. Timeout korrekt definiert

In diesem Fall ist der Timeout korrekt definiert, und der SKX Advance erkennt die zwei gesendeten Frames.

Bitte beachten: Um den Timeout korrekt zu definieren, muss dies unter Berücksichtigung der Pause zwischen gesendeten Frames des externen Geräts geschehen. Wie im Beispiel gezeigt wird , kann ein schlecht definierter Timeout (zu lang) zu Kommunikationsproblemen führen.

Beispiel: *Nachfolgend wird ein praktiches Beispiel aufgezeigt, bei dem es nötig ist den Timeout einzustellen. Der SKX Open (rechts) sendet bei Empfang eines Befehls des SKX Advance (links) am RS232-Port, ein Bestätigungstelegramm (ACK) und den Status in einem Abstand von 60 ms. Ist der Timeout im SKX Advance grösser als 60 ms gewählt, so wird dieser keines der Telegramme mitbekommen, und so seinen Status nicht aktualisieren.*

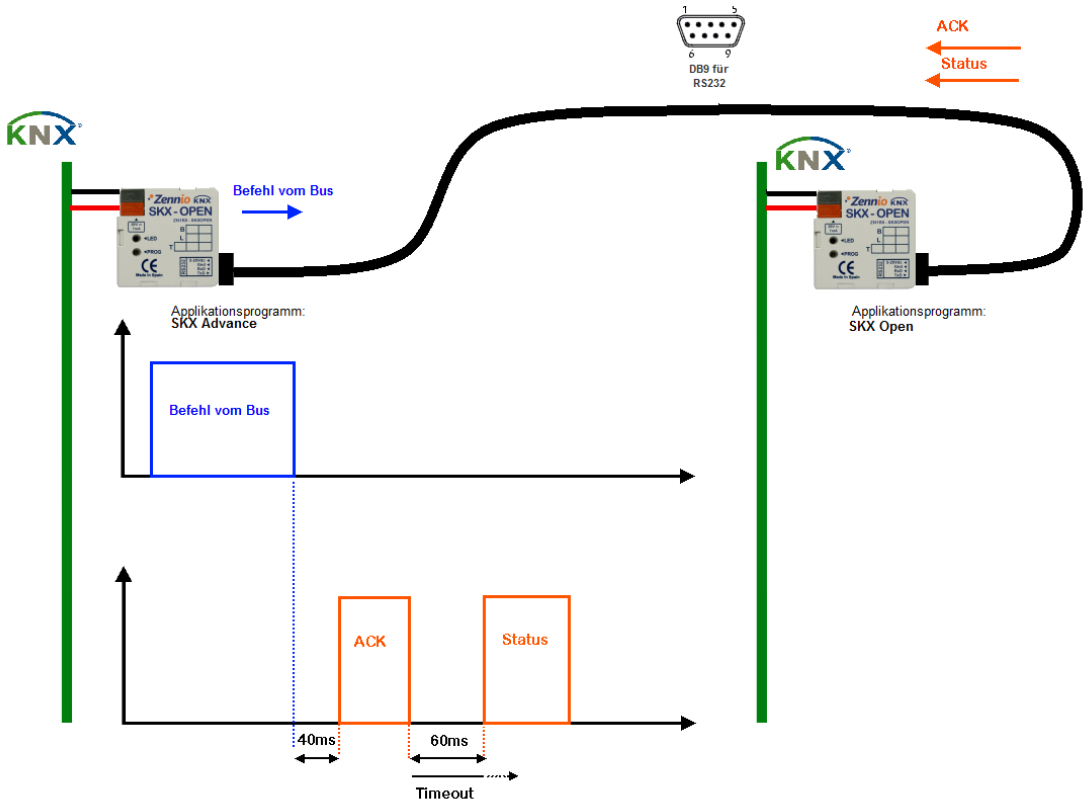


Bild 6. Beispiel eines SKX Advance mit einem SKX Open

3.2. KOMMUNIKATIONSKAPAZITÄT

Es müssen die Gruppen der Komunikationsobjekte welche benutzt werden sollen, freigegeben werden.

Grösse	Anzahl der Gruppen:	Anzahl der Objekte je Gruppe:
1 bit	4	10
1 byte	2	10
14 byte	1	5

Auf diese Weise stehen als 1-bit Objekte die Nummern 0-39, als 1-byte Objekt die Nummern 40-59, und als 14-byte Objekte die Nummern 60-64 zur Verfügung. Nach Freigabe der Parametergruppe erscheinen die dazugehörigen Reiter.

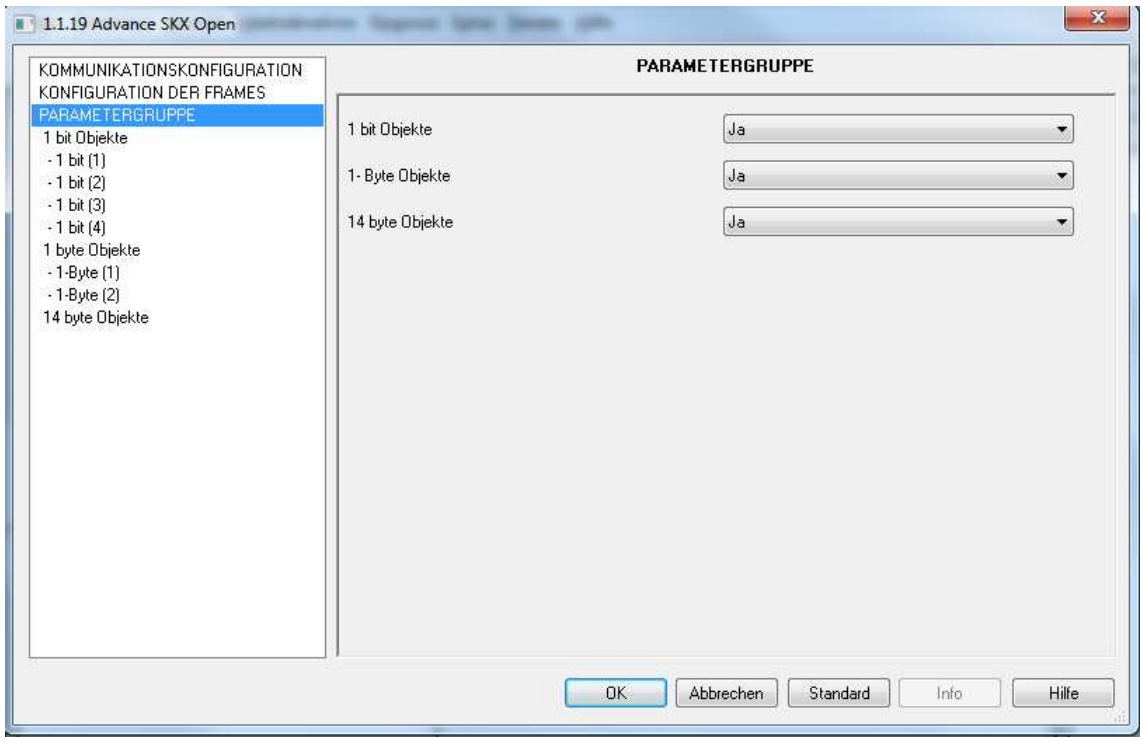


Bild 7. SKX Advance – Parametergruppen

Um die zur Verfügung stehenden Optionen konfigurieren zu können, müssen dann innerhalb jeder Gruppe von Kommunikationsobjekten jeweils die gewünschten Objekte freigegeben werden.

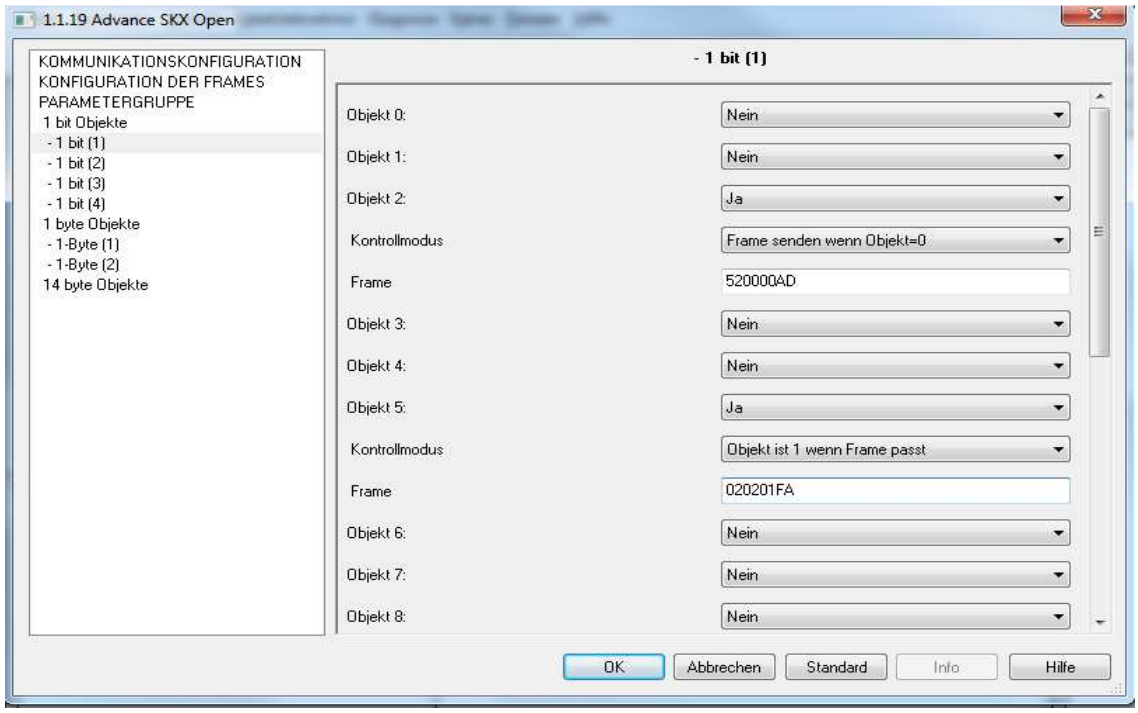


Bild 8. SKX Advance – Gruppe 1 der 1-bit Kommunikationsobjekte

Nach der Freigabe eines **1-bit Kommunikationsobjekt** stehen dann zwei Optionen zur Auswahl.

- **Obj X. Kontrollmodus:** Es stehen vier verschiedene Kontrollmöglichkeiten über Parameter zur Verfügung:

  Für die Komunikation KNX => RS232.

  - **Frame senden wenn Objekt = 0** Senden eines Frames (als Parameter in "Obj.X, Frame" eingegeben) an das über den seriellen Port integrierte Gerät, bei Empfang einer 0 auf dem Kommunikationsobjekt.
  - **Frame senden wenn Objekt = 1** Senden eines Frames (als Parameter in "Obj.X, Frame" eingegeben) an das über den seriellen Port integrierte Gerät, bei Empfang einer 1 auf dem Kommunikationsobjekt.

  Für die Komunikation RS232 => KNX

  - **Objekt ist 0 wenn Frame passt:** Senden einer 0 über das Kommunikationsobjekt bei Empfang eines Frames über den seriellen Port, der mit dem als Parameter angegebenen Frame übereinstimmt.
  - **Objekt ist 1 wenn Frame passt:** Senden einer 1 über das Kommunikationsobjekt bei Empfang eines Frames über den seriellen Port, der mit dem als Parameter angegebenen Frame übereinstimmt.

- **Obj X. Frame**: Mit diesem Parameter werden die Frames definiert, die für die Kommunikation in Betracht kommen. Die eingegebenen Frames müssen folgende Voraussetzungen erfüllen:

  - Die benutzten Zeichen müssen hexadezimaler Natur sein (0-9, A-F).
  - Die Zeichen A-F müssen Grossbuchstaben sein.
  - Die Länge des Frames muss eine gerade Parität aufweisen. Zwei Zeichen pro 1-Byte Hexadezimalwert.
  - Die maximale Länge beträgt 10 Byte in Hexadezimal.

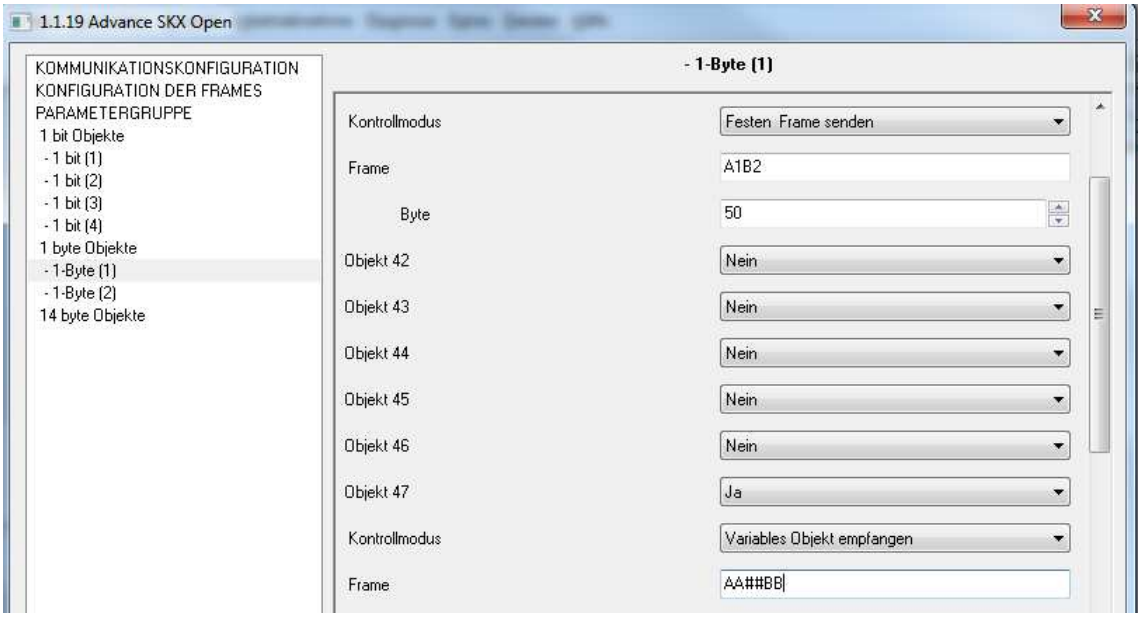


Bild 9. SKX Advance – Gruppe 1 der 1-byte Kommunikationsobjekte

Nach Freigabe eines **1-byte Kommunikationsobjekts** stehen dann die zwei Optionen zur Wahl des Kontrollmodus und der benutzten Frames für jedes Kommunikationsobjekt zur Verfügung:

- **Obj X. Kontrollmodus:** Es stehen sechs verschiedene Kontrollmöglichkeiten über Parameter zur Verfügung:

  Für die Komunikation KNX => RS232.

  - **Festen Frame senden:** Senden eines Frames (als Parameter in "Obj.X, Frame" eingegeben) an das über den seriellen Port integrierte Gerät, bei Empfang des parametrisierten Werts auf dem Kommunikationsobjekt.
  - **Variablen Frame senden:** Senden eines Frames (als Parameter in "Obj.X, Frame" eingegeben) an das über den seriellen Port integrierte Gerät, bei Empfang eines Werts auf dem Kommunikationsobjekt. Der gesendete Frame hängt vom empfangenen Objektwert ab.
  - **Variablen Frame senden (%):** Senden eines Frames (als Parameter in "Obj.X, Frame", eingegeben) an das über den seriellen Port integrierte Gerät, bei Empfang eines Werts auf dem Kommunikationsobjekt. Der gesendete Frame hängt vom empfangenen Objektwert ab.

  Für die Komunikation RS232 => KNX

- **Festes Objekt empfangen:** Senden eines parametrisierten Wert über das Kommunikationsobjekt bei Empfang eines Frames über den seriellen Port, der mit dem als Parameter angegebenen Frame übereinstimmt.

- **Variables Objekt empfangen:** Senden eines Werts über das Kommunikationsobjekt bei Empfang eines Frames über den seriellen Port, der mit dem als Parameter angegebenen Frame übereinstimmt. Der Wert ist abhängig vom empfangenen Frame

- **Variables Objekt empfangen (%):** Senden eines Werts über das Kommunikationsobjekt bei Empfang eines Frames über den seriellen Port, der mit dem als Parameter angegebenen Frame übereinstimmt. Der Wert ist abhängig vom empfangenen Frame

- **Obj X. Frame:** Mit diesem Parameter werden die Frames definiert, die für die Kommunikation in Betracht kommen. Die eingegebenen Frames müssen folgende Voraussetzungen erfüllen:

    - Die benutzten Zeichen müssen hexadezimaler Natur sein (0-9, A-F).
    - Die Zeichen A-F müssen Grossbuchstaben sein.
    - Die Länge des Frames muss eine gerade Parität aufweisen. Zwei Zeichen pro 1-Byte Hexadezimalwert.
    - Die maximale Länge beträgt 10 Byte in Hexadezimal.
    - Es werden Sonderzeichen wie ## für ein variables Byte, ** für null oder mehr Zeichen oder @1 oder @2 für die Subframes 1 und 2 benutzt.

Wird als Kontrollmodus die Option *Fester Frame senden* oder *Festes Objekt empfangen* gewählt, so erscheint eine dritte Option:

- **Obj X. Byte:** In diesem Feld wird das Byte definiert welches bei Empfang eines Frames gesendet werden soll, oder das erwartete Byte zum Senden eines definierten Frames.

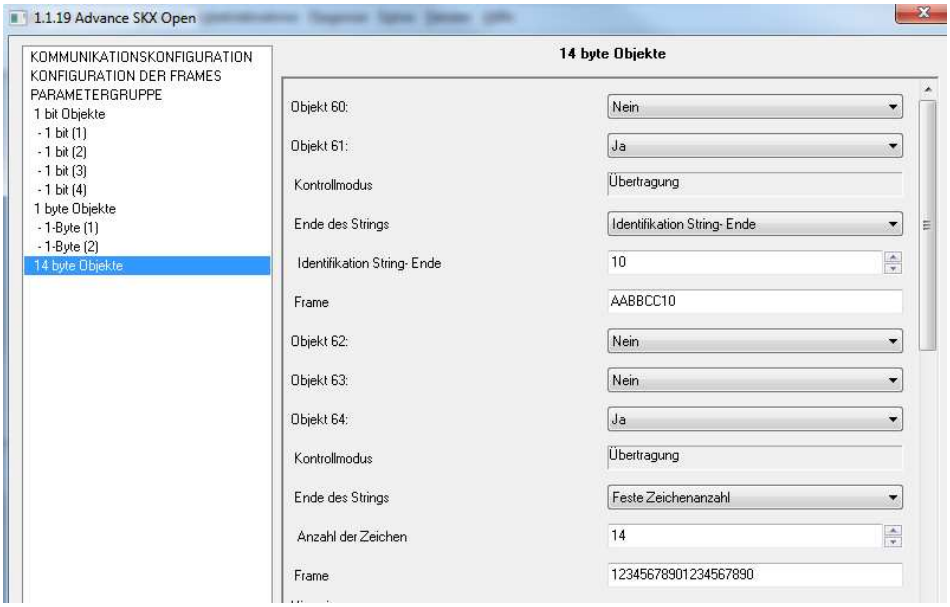


Bild 10. SKX Advance – Gruppe der 14-byte Kommunikationsobjekte

Als Letztes stehen die **14-byte Objekte,** welche über drei Optionen verfügen. Kontrollmodus, die Art der Kennzeichnung des String-Endes mit dem betreffenden Wert, und der benutzte Frame für jedes Kommunikationsobjekt.

- **Obj X. Kontrollmodus:** Es steht nur eine Kontrollmöglichkeit über Parameter zur Verfügung:

    - **Übertragung:** Übertragung eines vom seriellen Port empfangenen variablen Text-Strings auf den KNX-Bus, und umgekehrt..

- **Obj X. Ende des Strings:** In diesem Feld wird definiert wie das Ende des Text-Strings gekennzeichnet ist.

    - **Feste Zeichenanzahl:** hier wird eine konkrete Anzahl von Zeichen angegeben aus denen der Text besteht.

    - **Identifikation String-Ende:** hier wird das Byte innerhalb des Text-Strings definiert welches dessen Ende kennzeichnet.

- **Obj X. Frame:** Mit diesem Parameter werden die Frames definiert, die für die Kommunikation in Betracht kommen. Die eingegebenen Frames müssen folgende Voraussetzungen erfüllen:

    - Die benutzten Zeichen müssen hexadezimaler Natur sein (0-9, A-F).
    - Die Zeichen A-F müssen Grossbuchstaben sein.
    - Die Länge des Frames muss eine gerade Parität aufweisen. Zwei Zeichen pro 1-Byte Hexadezimalwert.
    - Die maximale Länge beträgt 10 Byte in Hexadezimal.
    - Es werden Sonderzeichen wie ## für ein variables Byte, ** für null oder mehr Zeichen oder @1 oder @2 für die Subframes 1 und 2 benutzt.

Bitte beachten: Es findet eine Überprüfung der eingegebenen Parameter über die Fehler-Kommunikationsobjekte statt. Dies Überprüfung wird bei Initialisierung des Gerätes durchgeführt.

Es wird darauf hingewiesen dass ein Hexadezimalcode von 2 byte, z.B. 0x2B 0x7F im Format "2B7F" in der ETS verarbeitet wird.

Der Benutzer verfügt über 20 Zeichen zur Definition eines Frames für einen seriellen Port. Dies erlaubt im Prinzip die Darstellung von 10 bytes. Zur Darstellung grösserer und vielseitiger Frames (variable Daten, Frames verschiedener Längen etc.) besteht die Möglichkeit verschiedene Frame-Abschnitte zu konfigurieren. Auf diese Art können Frames mit einer Länge von bis zu 29 byte verarbeitet werden.

3.3. KONFIGURATION DER FRAMES

Es besteht die Möglichkeit eine Reihe von speziellen Frames zu konfigurieren, welche eine umfassende Kommunikation mit dem zu integrierenden Gerät ermöglicht. Diese speziellen Frames sind folgende:

- **Kopfzeile:** Es kann festgelegt werden, ob dieser Frame automatisch an den Anfang eines jeden empfangenen und gesendeten Frames gesetzt wird, oder nur wenn dies durch die Spezialzeichen '@h' gekennzeichnet ist.

- **Fusszeile:** Ähnlich wie die Kopfzeile, nur dass sie an das Ende eines Frames gesetzt wird. Die dazugehörigen Spezialzeichen sind '@f'

- **Subframe (1 und 2):** Dies sind zwei optionale Frames die unter Benutzung von '@1' und '@2' in die Haupt-Frames eingefügt werden können.

- **Checksumme:** Das Applikationsprogramm ermöglicht die automatische Berechnung einer Reihe von Checksummen, sa dass diese in die Frames eingefügt werden können. Für den Fall, das eine automatische Einfügung gewünscht wird, so geschieht dies immer am Ende des Frames, nach der Fusszeile, falls existent. Die Spezialzeichen für eine manuelle Einfügung sind '@c'

- **Spezialzeichen:** Dank dem Einsatz dieser Zeichen kann die Komplexität der Frames erhöht werden und verschiedene Bytes die von einem zum anderen Frame variieren, ignoriert werden.

- **Bestätigung (ACK):** Der SKX verfügt über die Möglichkeit bei Empfang eines Frames vom RS232-Port, automatisch definierte Bestätigungstelegramme zu senden. Ferner besteht die Möglichkeit der Konfiguration eines speziellen Frames als Antwort, den das externe Gerät sendet. Wird dieser vom SKX empfangen, so findet das Senden des ACK-Frames nicht statt.

3.4. SPEZIALZEICHEN

Es werden eine Reihe von Spezialzeichen definiert, welche an die vom Benutzer definierten Frames angehängt werden können, um ihnen auf diese Weise eine grössere Vielseitigkeit und die Möglichkeit umfangreicherer Befehlskonstruktionen zu verleihen.

- **?1, ?2, …, ?9:** bedeuten dass die folgenden n bytes bei der Analyse des Frames ignoriert werden.
- **@h, @f:** Aufnahme der Kopf- bzw. Fusszeile in den Frame.
- **@1, @2:** Aufnahme des Subframes 1 bzw. 2 .
- **@c:** Aufnahme des/der Checksummen-Bytes
- ****:** Existenz von null oder mehr Byte beliebigen Werts vor definiertem Teil (ausser @h, @f, @1 y @2). Die Zeichen ** stellen also die minimale Zeichenanzahl dar, die vor dem konstanten Teil eines Frames zu finden sind.
- ##: Erscheinung eines variablen Bytes oder Text-Strings innerhalb eines Frames. Im speziellen Fall der Benutzung eines 14-byte Objekts, muss entweder nach dem Text-String das entsprechende End-Byte erscheinen, oder der String muss aus einer exakt konfigurierten Anzahl von Zeichen bestehen.
  - Nachfolgend einige Beispiele von definierten benutzerspezifischen Frames.
  - Kopfzeile: FF; Fusszeile: EE; Subframe 1: DD; Subframe 2: CC

Frame	***Beschreibung***	***Gültige Beispiele***
@h AABB##CC @f @c	Aufnahme einer Kopfzeile, zwei festen Byte, ein variables Byte, ein zusätzliches Byte, Fusszeile und Checksumme	FF AA BB 01 CC EE SS, wobei SS die Checksumme vom ersten bis zum vorletzten Byte repräsentiert. Es wurde der Wert 01 über das entsprechende 1-byte Objekt empfangen.
AA ?3 BB	Die akzeptierten Frames müssen eine Länge von 5 byte aufweisen, wobei das erste und letzte Byte AA bzw. BB sind.	AA 01 02 03 BB AA 04 04 04 BB AA AA AA AA BB
AA ** BB CC	Die Länge dieser Frames ist variabel, da das Symbol ** enthalten ist, welches die mögliche Erscheinung mehrerer Zeichen bedeutet.	AA BB CC AA 01 02 BB CC AA 01 BB 01 CC 01 BB CC → Ungültig
@1 @2 ##	Erscheinung der vorher definierten Subframes an erster Stelle, danach ein variables Byte oder ein variabler Text-String	DD CC 01: Obj = 01 DD CC 02: Obj = 02 DD CC ‘h’ ‘o’ ‘l’ ‘a’ ‘\0’: Obj = hola

3.5. DIE FEHLEROBJEKTE

Das Objekt Fehler gibt an, dass ein Fehler vorliegt. Dieser wird dann durch die weiteren Objekte konkretisiert. Jedes Objekt hat eine konkrete Bedeutung:

- **Fehler: Länge ungerade** Die Länge des Frames weist eine ungerade Zeichenanzahl auf. Die mittels ETS eingegebenen Frames müssen immer eine gerade Zeichenanzahl aufweisen.

- **Fehler: falsche Nutzung von '*' oder '?"** Fehler in der Benutzung von '*' oder '?' Es befindet sich kein konstanter Wert nach '**' oder ein inkorrektes Zeichen nach '?'

- **Fehler: falsche Nutzung von '@'** Fehler in der Benutzung von '@' Das Zeichen nach '@' ist nicht korrekt.

- **Fehler: falsche Checksumme** Es liegt ein Fehler bei der Berechnung der Checksumme vor Es gibt nichts in der Checksumme zu berechnen, da z.B. der Frame nur mit '@c' definiert wurde, oder der Offset der Checksumme ist zu gross

- **Fehler: falsche Nutzung von '#'** Fehler in der Benutzung von '#' Fehler in der Syntax von '##', oder es ist nicht möglich '##' einem variablen Wert zuzuordnen (z.B. bei den 1-bit Objekten)

- **Fehler: zu lang** Der empfangene Frame ist länger als erlaubt. Die maximale Länge der Frames beträgt 29 byte

- **Fehler: Empfang** Fehler beim Empfang am seriellen Port. Unbestimmter Empfangsfehler am seriellen Port

- **Fehler: nicht hexadezimal** Fehler eines nicht hexadezimalen Zeichens. In irgendeinem definierten Frame wurde ein hexadezimales Zeichen erwartet [0-9][A-F], und ein andersartiges erhalten.

Nach einem Reset oder einer Neuprogrammierung führt das System eine Überprüfung auf Parameterfehler durch:

- Analyse des vom Benutzer definierten End-Bytes; enthält dieses keinen Hexadezimalwert, wird das entsprechende Fehlerobjekt gesendet.
- Analyse der Kopfzeilen, Fusszeilen, Subframes, etc. Alle erkannten Fehler werden simultan über die entsprechenden Objekte gemeldet.
- Individuelle Analyse der vom Benutzer definierten Frames. wird ein Fehler erkannt, so wird er über das betreffende Objekt gemeldet.

Nach dieser Überprüfung wird das allgemeine Fehlerbit auf den Bus gesendet, zur indikation der Existenz einer Störung.

Auf der anderen Seite können auch Fehlermeldungen während der Kommunikation oder bei späteren Überprüfungen vorkommen, wenn der empfangene Frame mit dem vom benutzer definierten verglichen wird.

3.5.1. FEHLERBEISPIELE

- **Länge ungerade:** Wird versucht einen Frame mit ungerader Zeichenanzahl zu senden, so wird eine 1 über das Objekt 65 gesendet. Ausserdem wird eine 1 über das Objekt 66 gesendet (Länge ungerade).

***Beispiel**: In diesem Fall wurde für das Objekt 0 der Frame 520001A definiert (ungerade Zeichenanzahl) Dieser Fehler wird über das entsprechende Objekt auf den Bus gesendet.*

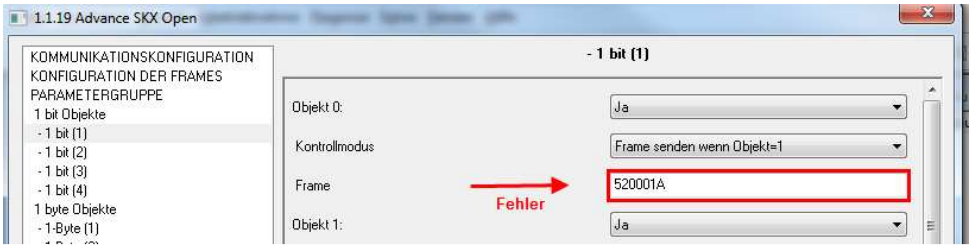


Bild 11. Ungerader Frame

- **Falsche Nutzung der Spezialzeichen ** oder ?** Falls nach einem Spezialzeichen ? statt einer Ziffer irgendein anderes Zeichen erscheint, so wird ausserdem eine 1 über das Objekt 67 gesendet.

***Beispiel**: In diesem Fall wurde innerhalb eines 1-byte Objekts das Senden eines Frames A1##?AB2 definiert (ungültiger Wert für dieses Spezialzeichen) Dieser Fehler wird über das entsprechende Objekt auf den Bus gesendet.*

Bild 12. *Falsche Nutzung des Spezialzeichens '?'*

- **Fehler in der Benutzung von '@'** Das Spezialzeichen ermöglicht die Aufnahme von Kopfzeilen, Subframes, Fusszeilen oder Checksummen in einen Frame. In diesem Fall können nur @h,@f,@1,@2 oder @c benutzt werden, bei unterschiedlichen Zeichen wird ausserdem eine 1 über das Objekt 68 gesendet.

***Beispiel**: In diesem Fall wurde in den Frame für das 1-bit Objekt 1 '@3' eingefügt (ungültiger Wert für dieses Spezialzeichen), daher wird eine 1 über das Objekt 68 gesendet.*

Bild 13. ***Falsche Nutzung des Spezialzeichens '@'***

- **Fehler bei der Berechnung der Checksumme:** Die Berechnung der Checksumme liefert ein falsches Ergebnis, da die Operation mit einem leeren Frame durchgeführt wird oder der Offset zu gross konfiguriert wurde, was ein Fehlertelegramm auf Objekt 69 zur Folge hat.

***Beispiel**: Im Beispiel wurde versucht die Checksumme zu kalkulieren, obwohl der Frame als leer definiert wurde.*

Bild 14. ***Leerer Frame mit Fehler in der Checksummenberechnung***

- **Fehler in der Benutzung von '##'** Jedes Mal wenn dieses Spezialzeichen, welches die Aufnahme eines variablen Werts ermöglicht, falsch benutzt wird, hat dieses ein Fehlertelegramm über Objekt 70 zur Folge

***Beispiel**: Werden die Zeichen ## bei einem 1-bit Objekt (in diesem Fall Objekt 0) benutzt, so wird ein Fehlertelegramm über Objekt 70 gesendet, da ein variabler Teil in einen Frame aufgenommen wurde, für den dies nicht zulässig ist.*

Bild 15. ***Frame mit einer fehlerhaften Benutzung des Spezialzeichens ##***

- **Falsche Länge, Frame länger als 29 byte:** Aufgrund der Tatsache dass Kopfzeilen, Fusszeilen und Subframes Längen von x Byte erlauben, besteht die Gefahr Frames länger als 29 Byte zu definieren. Für den Fall dass dieses passiert, wird eine 1 über das Objekt 71 gesendet.

***Beispiel**: Wird eine Kopfzeile von 10 byte, eine Fusszeile von weiteren 10 byte und ein Subframe von 4 byte definiert, so wird bei Aufnahme weiterer 10 byte die Höchstgrenze überschritten, und eine 1 über das Fehlerobjekt 71 gesendet.*

Bild 16. ***Frame mit einer Länge grösser als 29 Byte***

- **Fehler bei den Kommunikationsparametern** Die Konfiguration der seriellen Kommunikationsparameter stimmt nicht mit der Konfiguration der empfangenen Frames überein. (Unterschiedliche Geschwindigkeit, Parität, Stopbits,....) In diesem Fall wird eine 1 über das Objekt 72 gesendet

Beispiel: *Das externe Gerät verfügt über eine definierte Geschwindigkeit von 9600 Baud und der SKX von 1200 Baud. Alle Fehler die die Kommunikation betreffen, werden über das Objekt 72 "Fehler:Empfang" gemeldet.*

- **Fehler eines nicht hexadezimalen Zeichens:** Wurde vom Benutzer ein nicht hexadezimales Zeichen in den SKX-Parametern eigegeben, so hat dies ein Fehlertelegramm über Objekt 73 zur Folge. Kleinbuchstaben werden ebenfalls als Fehler angesehen.

Beispiel: *Der eingegebene Wert für das 1-bit Objekt 0 ist ABcD Da in diesem Fall ein Kleinbuchstabe eingegeben wurde, wird eine 1 über das Objekt 73 gesendet.*

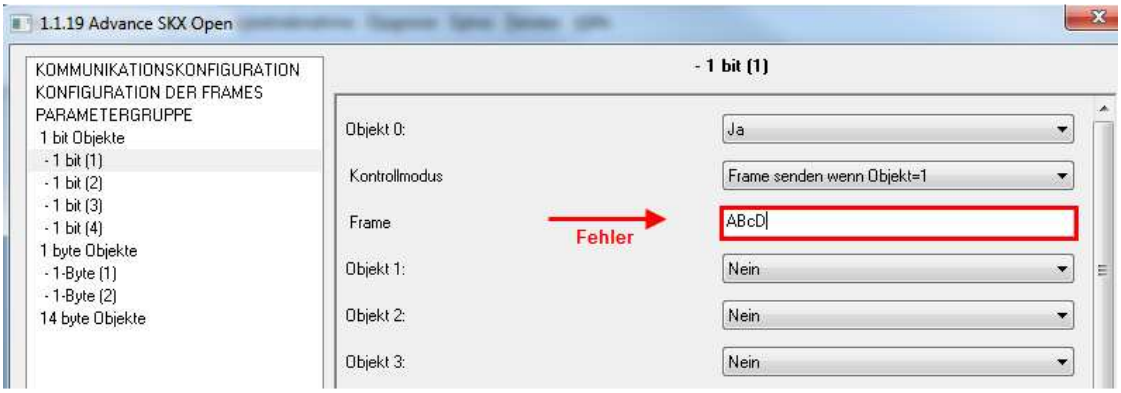


Bild 177. ***Kleinbuchstabe***

Bitte beachten: Wird ein kompletter oder partieller Download durchgeführt, so wird im Falle eines Konfigurationsfehlers eines Frames (ungerade Zeichenanzahl, exzessive Länge, nicht-hexadezimale Zeichen) unmittelbar die entsprechende(n) Fehlermeldung(en) gesendet.

ZENNIO TECHNISCHE DOKUMENTATION

WERDE BENUTZER!

http://zennio.zendesk.com

TECHNISCHER SUPPORT