



SKX LINKER

SKX OPEN

ZNRX – RS232



Edition 1.0

1. Introduction.....	3
1.1. SKX Linker.....	3
1.2. Application Program: SKX Open.....	3
1.3. SKX Open Overview	4
2. Installation	5
2.1. SKX Linker installation. KNX Bus	5
2.2. SKX Linker connection to RS232 port.....	5
3. Parameterization	6
3.1. Main Setup Tab.....	7
3.2. Parameter Groups.....	10
3.3. Error Code Communication Object.....	12
3.3.1. Possible Errors.....	12
3.3.2. Example of Errors.....	13

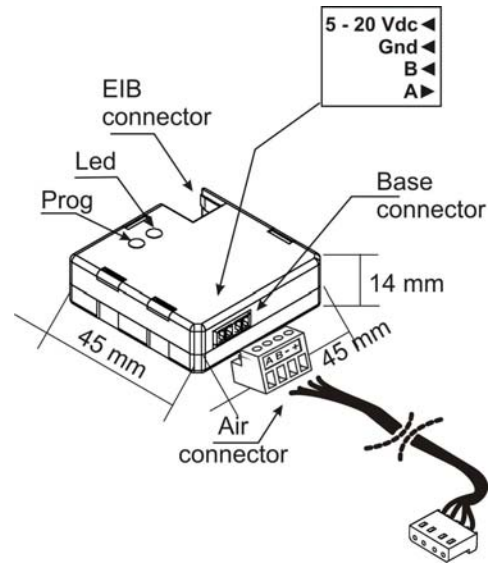
1. INTRODUCTION

1.1. SKX LINKER

The SKX Linker is the Zennio device that allows connecting the bus KNX with the RS232 data bus.

Device Features:

- Reduced size: 45 x 45 x 14mm.
- Designed to be placed either in a wiring box or inside electrical panels.
- Proprietary communication protocol
- Several communication speeds and error correction mechanisms.
- Ideal for M2M applications.
- Based in an EIB/KNX BIMM112 core.
- Total data saving.
- Following the CE guidelines



Description of the elements:

- **Prog:** it is the button used to enter the device in Programming Mode. When pushing it at the beginning, after supplying the bus with power, the device is forced to enter in Safe Mode.
- **Led:** it is the light signal that indicates that the device is in Programming Mode. When the device enters in Safe Mode, it blinks with a 0,5 seconds-long cycle time.

1.2. APPLICATION PROGRAM: SKX OPEN

The goal of this manual is explaining the specific application program developed in order to connect external devices that can be controlled via a RS232 port to KNX.

The SKX Open is a product for communication between KNX and serial protocol RS232 in a totally free environment, since it is able to control any device via RS232 independently of the hexadecimal code that the device generates. This is possible by associating hexadecimal codes with communication objects in the SKX Open configuration in ETS.

Note: Any hexadecimal code can be integrated but some length restrictions for them must be bear in mind in SKX Open configuration (additional Information in the section “3.2 Group of parameters”).

Communication between RS232 interface and KNX bus takes place in the SKX Linker device, which allows exchanging information in a bidirectional way. Thus, data from the KNX bus can be used to control a device linked to the RS232 interface or, the opposite, it allows controlling KNX devices through 1-bit data typed communication objects with data from the external device received through the RS232 interface.

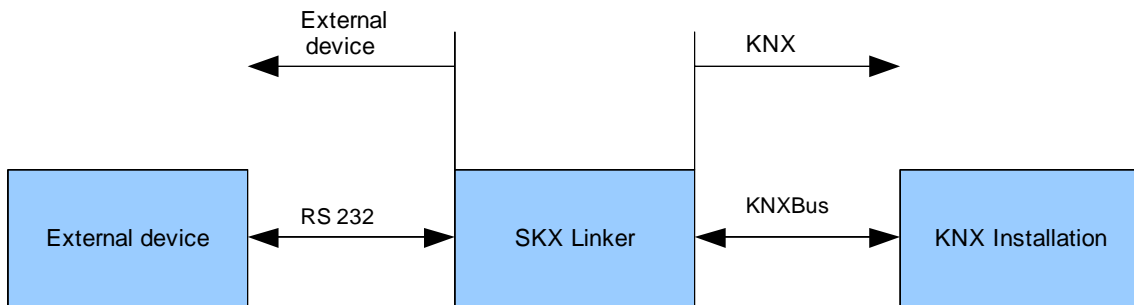


Figure 1. SKX Linker Bidirectional Communication

1.3. SKX OPEN OVERVIEW

The SKX Open presents a series of features that are configurable in the ETS:

- Transmission speed: (1200, 2400, 4800, 9600, 19200)
- End of frame recognition with Timeout or End Frame Byte.
- 44 communication objects
- Error detection: 1 byte – error code with bit mask.
- Maximum frame length: 10 bytes / 20 HEX characters (without including End Frame Byte, if exists).

2. INSTALLATION

2.1. SKX LINKER INSTALLATION. KNX BUS

The SKX Linker installation is performed the same way that any other KNX device. We just need to connect the device to KNX bus through the specific KNX connector and it will be ready to be programmed.

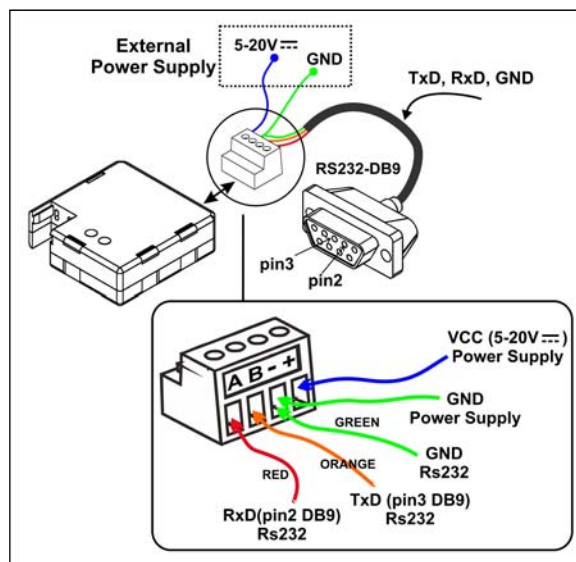
Once the device is provided with power supply from the bus, both, the physical address and the SKX Open application program can be downloaded.

This device does not need external supply, since it just works with the power supply of KNX bus. Nevertheless, it will be necessary to add an external power supply to the RS232 bus, independent of the KNX bus power supply, as defined in its standard.

2.2. SKX LINKER CONNECTION TO RS232 PORT

Connection to RS232 port is performed using an unfixed specific wire clamp in SKX Linker, which makes easier its connection and installation.

In the following figure, the SKX Linker connection to both protocols (RS232 and KNX) is shown:



SKX Linker wire Clamp	BUS RS232
A	RSA
B	RSB
-	GND
+	+12V

Figure 2. SKX Linker Connection

3. PARAMETERIZATION

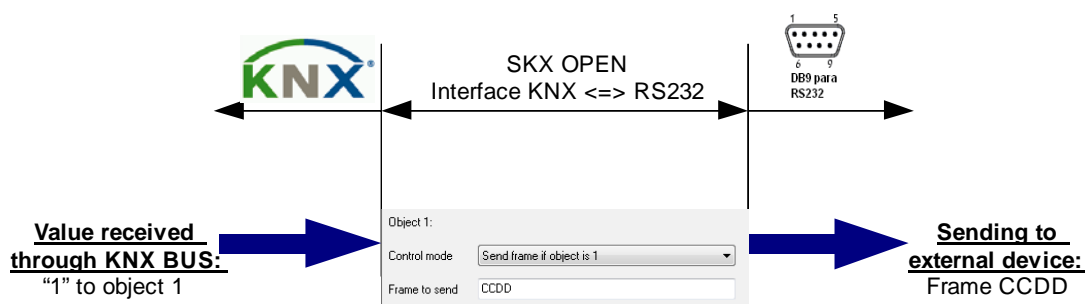
As it was seen in the previous sections, the device used for the application program SKX Open already existed in Zennio product catalogue, the ZN1RX-RS232 or SKX Linker. Thanks to this application program, any device with a RS232 interface can be integrated with the KNX system if the hexadecimal codes used for the orders are known.

The SKX Open provides 48 1bit-typed communication objects, which are used for interaction between KNX and RS232. Moreover, an additional 1 byte-typed communication object for Error Control provides information about the problems that come up when working (non hexadecimal values in parameterization, lower case letters, odd hexadecimal message length in parameterization, the message received through the serial port is too long or it contains errors, any error happened in the last sent message or it happened when comparing the parameterized value with the received one).

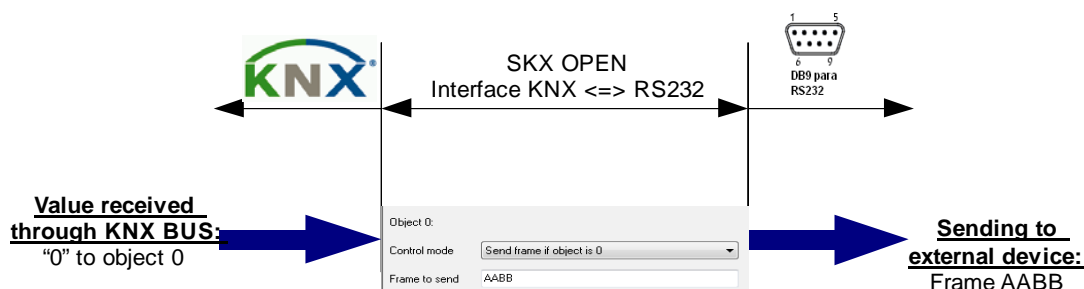
There are four options of controlling each communication object through the parameters:

KNX => RS232 Communication. Switching:

- **Sending frame** (that is configured as a parameter) to the integrated device through the serial port when **receiving a 1** through the communication object in KNX bus.

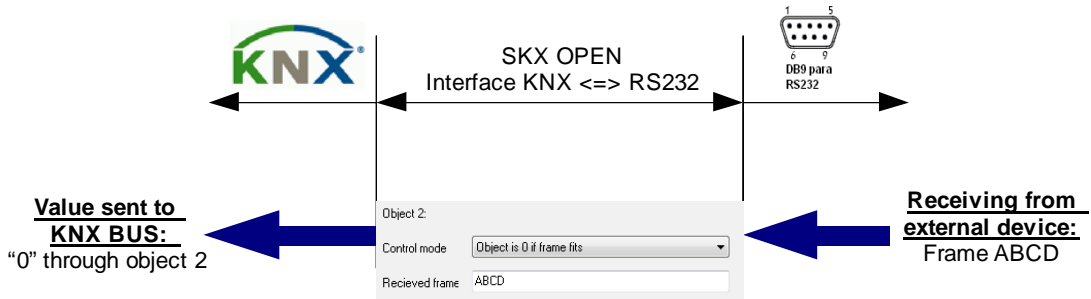


- **Sending frame** (that is configured as a parameter) to the integrated device through the serial port when **receiving a 0** through the communication object in KNX bus.

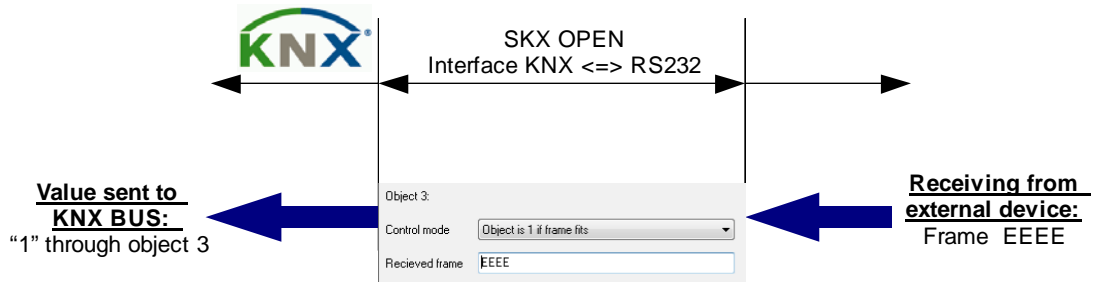


RS232 => KNX Communication. Switching:

- **Sending a 0 value** to the KNX Bus through the communication object in case the received frame via the serial port fits the one configured as a parameter.



- **Sending a 1 value** to the KNX Bus through the communication object in case the received frame via the serial port fits the one configured as a parameter



3.1. MAIN SETUP

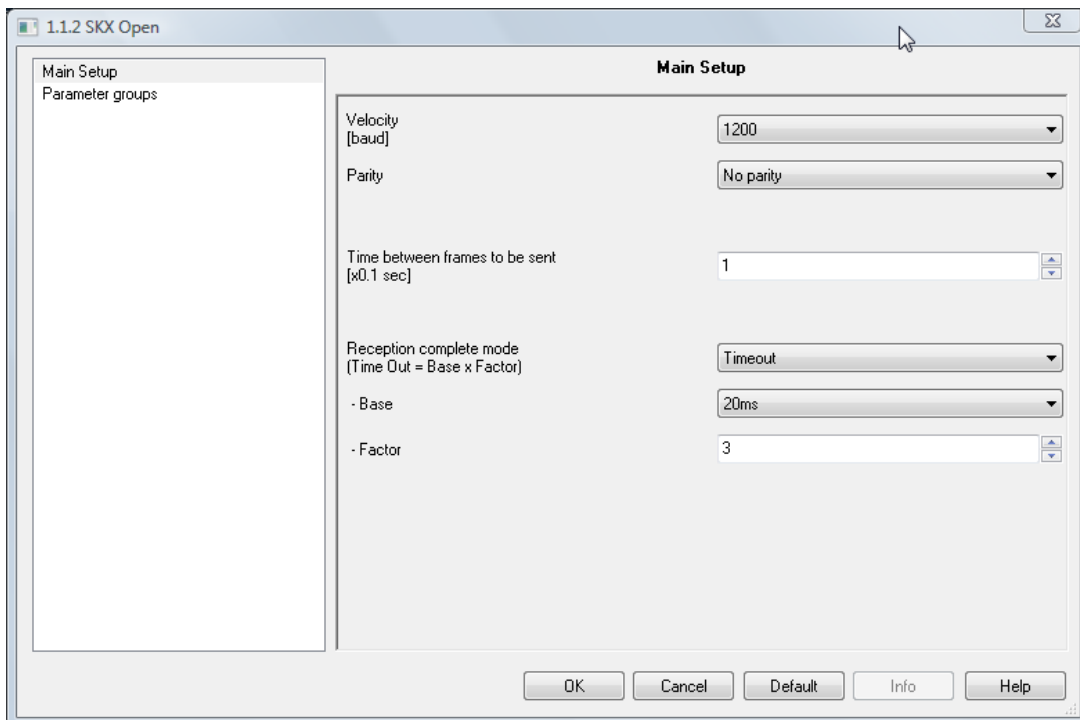


Figure 3. SKX Open – Main Setup

In this tab the parameters regarding the physical communication are configured.

- **Velocity:** 1200-2400-4800-9600-19200
- **Parity:** No parity, even parity, odd parity
- **Time between frames to be sent:** It is the configurable delay time when sending frames towards the serial port. It allows linking more than one communication object to the same group address, as the SKX Open will be able to send several frames coming from different communication objects in an ordered way, which means a perfect acquiring and interpretation of data in reception. This parameter will depend on the receptor device.
- **Reception complete mode:** In order to detect the frame ending, there are two choices:
 - **Timeout:** it is the time while the received frame is being waited since the last bit was received, so the frame is meant to end after that time.
 - **End frame byte:** the end of frame can be delimited by a specific byte. Any time the end frame byte is received by the SKX Linker, it recognizes the end of frame. In this case there is a security timeout in reception that can expire if the end frame byte is not received. When receiving a frame that is longer than 10 bytes, it will be ignored and a communication object Error Code will be sent to the KNX bus in order to inform about it.

Example: An external device takes 80ms to send the complete frame.

First case: The user configures 200ms for Timeout. Supposing the external device wants to send a second frame immediately after the first one, the functioning will be the one represented by the following figure:

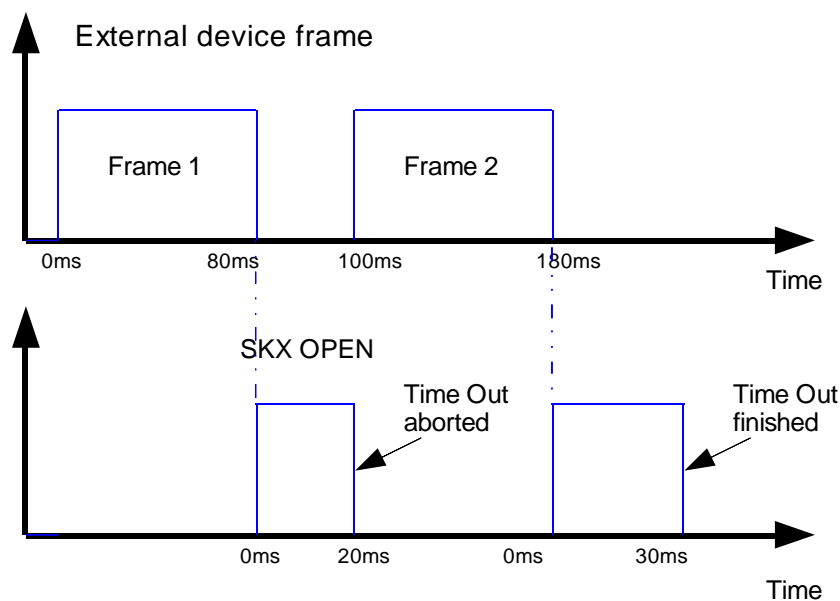


Figure 4. Too long Timeout

The SKX Open restarts its Timeout when the frame 1 ends, but before it expires, the frame 2 is received and thus, the Timeout is restarted again when frame 2 finishes. This way, the Timeout just expires after frame 2 reception and the SKX Open considers all the information that has been received until that time (frame 1 and frame 2) as a single frame and therefore, the SKX Open will not recognize any of them, due to a too long Timeout value.

Second case: The user configures 60ms for Timeout. The functioning will be the one represented by the following figure:

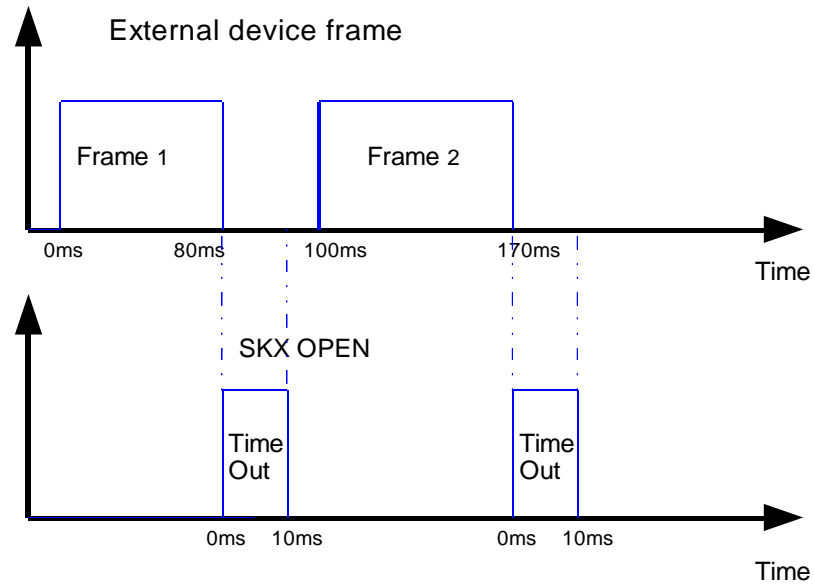


Figure 5. Too short Timeout

In this case, the external device has not enough time to send the frame before the Timeout ends. The SKX Open does not receive the complete frame and thus, it does not recognize it. This is due to the too short Timeout.

The Timeout must be defined according to the time between frames in the external device transmission. If it is not taken into account and the Timeout is too long or too short, frames reception could have errors.

Example: This is a real case where the Timeout value must be correctly configured for a right reception of the frames. When the SKX Linker (at right) receives an order from SKX Open (at left) through RS232 port, it sends an acknowledge frame (ACK) and a Status frame, separated by 60ms time. If the Timeout value in SKX Open is longer than 60ms, the SKX Open will not detect any of the frames, and thus, will not update the status in SKX Open.

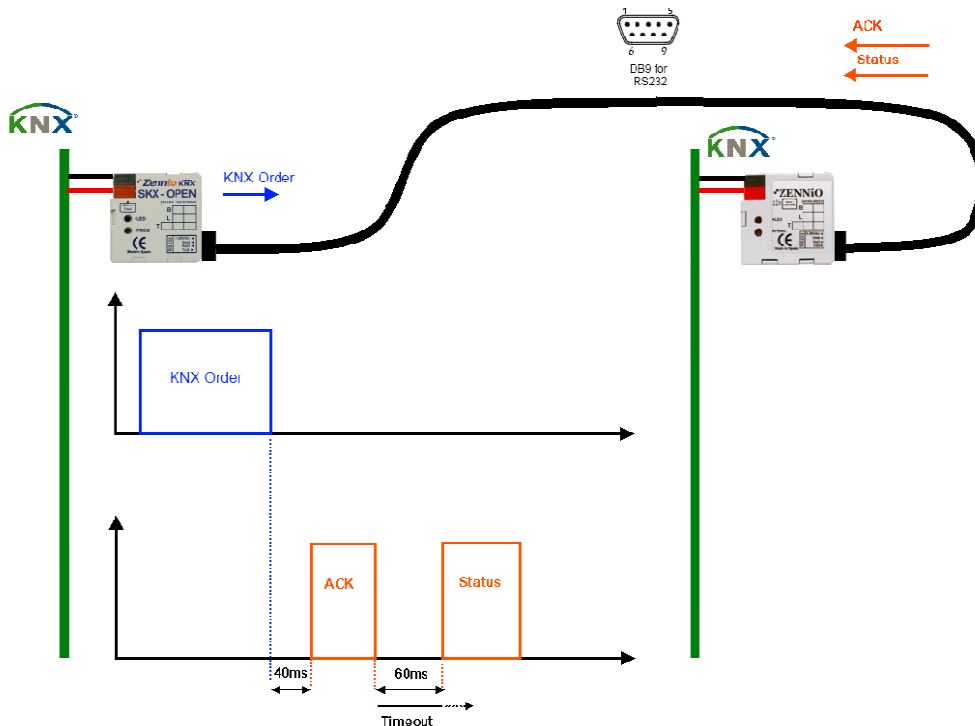


Figure 6. Example with SKX Open and SKX Linker

3.2. PARAMETER GROUPS

In this tab, the group of communication objects that we need must be enabled. There are 4 groups with 12 1 bit-typed communication objects each.

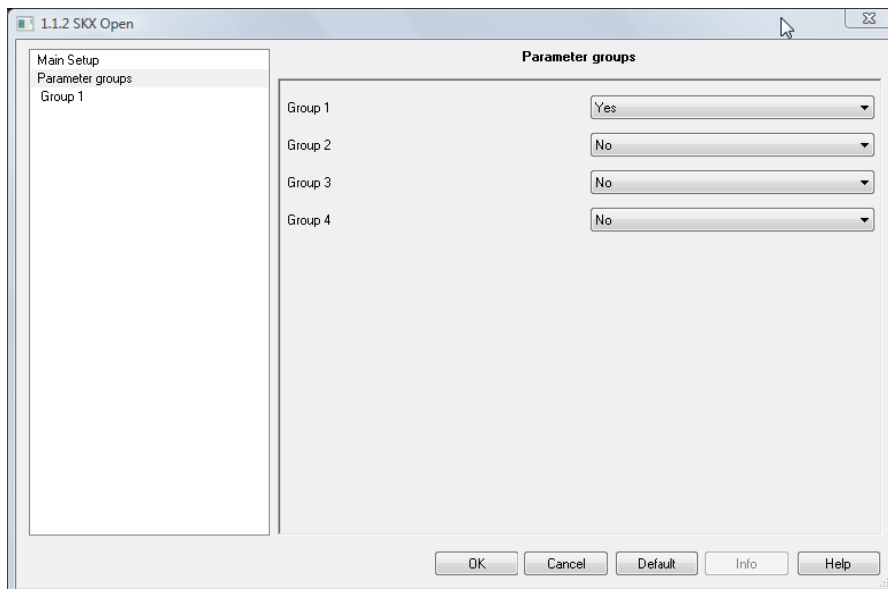


Figure 7. SKX Open – Parameter Groups

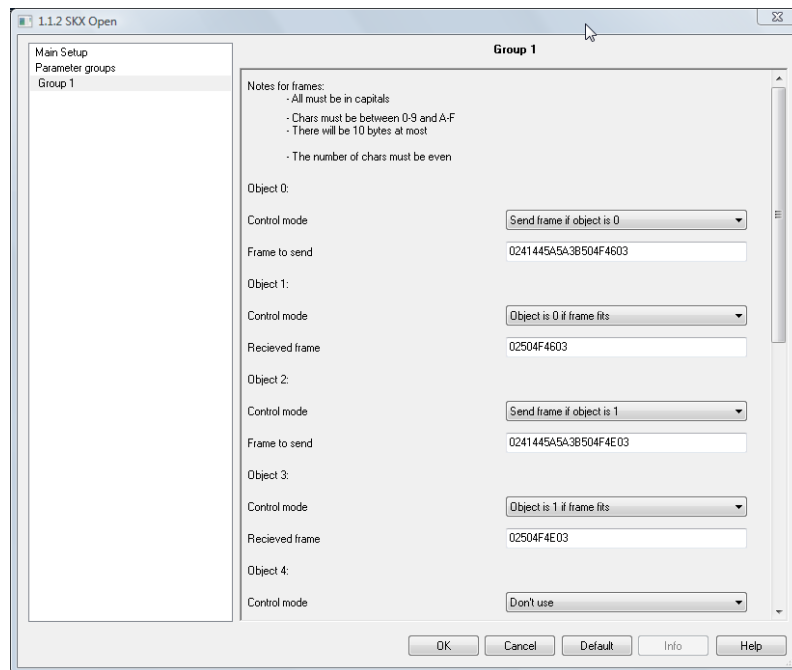


Figure 8. SKX Open – Group X

In **Group X** tab (there is one tab for each enabled group), the control mode and frames to be sent/received are selected for each communication object:

- **Obj X. Control mode:** This parameter allows defining if the communication object is used for a KNX to RS232 or RS232 to KNX communication, as well as the effects of the control:

For KNX => RS232 Communication

- **Send frame if object is 0:** Send the frame (configured in the parameter “Obj X. Frame to send”) to the external device through the serial port when the value of the communication object is 0.
- **Send frame if object is 1:** Send the frame (configured in the parameter “Obj X. Frame to send”) to the external device through the serial port when the value of the communication object is 1.

For RS232 => KNX Communication

- **Object is 0 if frame fits:** Send a 0 value of the communication object to the KNX bus when receiving a frame that fits the one that is set as a parameter “Obj X. Received frame” through the serial port.
- **Object is 1 if frame fits:** Send a 1 value of the communication object to the KNX bus when receiving a frame that fits the one that is set as a parameter “Obj X. Received frame” through the serial port.

According to the chosen control mode (KNX => RS232 or RS232 => KNX) we need to configure the corresponding parameter in each case:

🟡 **Obj X. Frame to send or Obj X. Received frame:** In this field we set the frames that must be taking into account by SKX Open for communication KNX => RS232 or RS232 => KNX respectively. The strings that we input in these fields must fulfill the following requirements:

- The characters must correspond to hexadecimal values (0-9, A-F).
- The characters A-F must be input in upper case letters.
- The frame length must be an even, as two characters correspond to 1byte hexadecimal value.

***Note I:** There is a communication object Error Code for controlling the string that the user configures in this tab. This error control takes place during set-up.*

***Note II:** If we want to use a 2 bytes-length hexadecimal frame, such as “0x2B0x7F”, it must be input in the ETS parameters with the format “2B7F”.*

3.3. ERROR CODE COMMUNICATION OBJECT

Number	Name	Object Function	Description	Grou...	Length	C	R	W	T	U	Data Type	Priority
48	Error code			0/0/3	1 Byte	C	R	-	T	-		Low
3	Object 3			0/0/2	1 bit	C	-	W	T	-		Low
2	Object 2			0/0/1	1 bit	C	-	W	T	-		Low
1	Object 1			0/0/2	1 bit	C	-	W	T	-		Low
0	Object 0			0/0/1	1 bit	C	-	W	T	-		Low

Figure 9. Communication object – Error code

The 1byte-typed communication object indicates the current problems in the communication using an accumulative code or bit mask. Each bit of the communication object value has a special meaning:

- **Bit 0.** No hexadecimal character in any configured frame
- **Bit 1.** Upper case character in any configured frame
- **Bit 2.** Odd length in any configured frame
- **Bit 3.** The error, which is defined with the rest of the bits, is in the current frame.
- **Bit 4.** There is an error in the RS232 port communication, such as velocity, parity, frame length...
- **Bit 5.** The received frame is longer than 10bytes.

3.3.1. POSSIBLE ERRORS

The communication object Error Code is updated and sent to the KNX bus in the following cases:

Error Definition	N° of error (in setup and when it does not happen in the current frame)	N° of error when it happens in the current frame
Error: Empty string in frame parameter	NO	09h (bit0 and bit3)
Error: End Frame Byte is not received	NO	28h (bit5 and bit3)
Error: Communication parameters	18h (bit4 and bit3)	18h (bit4 and bit3)
Error: No hexadecimal value	01h (bit0)	09h (Bit0 and bit3)
Error: Odd length value	04h (bit2)	0Ch (bit2 and bit3)
Error: Lower case character in frame	02h (bit1)	0Ah (bit1 and bit3)
Error: Frame length	NO	28h (Bit5 and Bit3)

3.3.2. EXAMPLE OF ERRORS

- 🔴 **Empty string in frame parameter:** In case an object is enabled and the frame field in parameterization is empty an error with code 09 will take place.

Example: In this case, when the object value is 0, this error will take place.

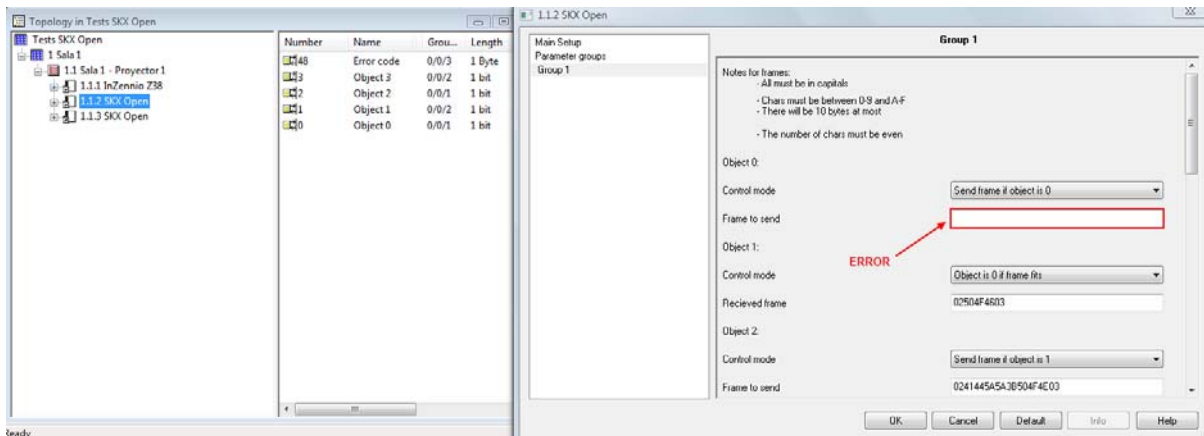


Figure 10. Empty string in frame parameter

#	Time	Service	Flags	Prio	Src.addr	S...	Dest.addr	Destination	Rout	DPT	Type	Data
15...	12:19:11.374	to bus		S	15.15.255	N...	1.1.2		6	-	ACK (S=12)	
15...	12:19:11.430	to bus		S	15.15.255	N...	1.1.2		6	-	T_Disconnect	
15...	12:19:51.023	to bus		L	15.15.255	N...	0/0/1	Cinema ON / OFF	6	1 bit	Write	S00
15...	12:19:51.042	from bus		L	1.1.2		0/0/3	Error projector	6	1 byte	Write	S09 14 %

Figure 11. Error code when there is an empty string in frame parameter

- 🔴 **End Frame Byte is not received:** If the detection of the frame end is an End-Frame Byte and it is not received when the timeout expires or there is an error in the received value, and thus it is not detected, an error with code 28 will take place.

Example: If the user defines an End Frame Byte FF, the frame reception will be successful if the byte FF is received. If the external device sends 02ADFA (because of a transmission error) or it sends 02ADFF but it takes more than 60ms (in our example), the SKX Open will send the error code 28 to the bus.

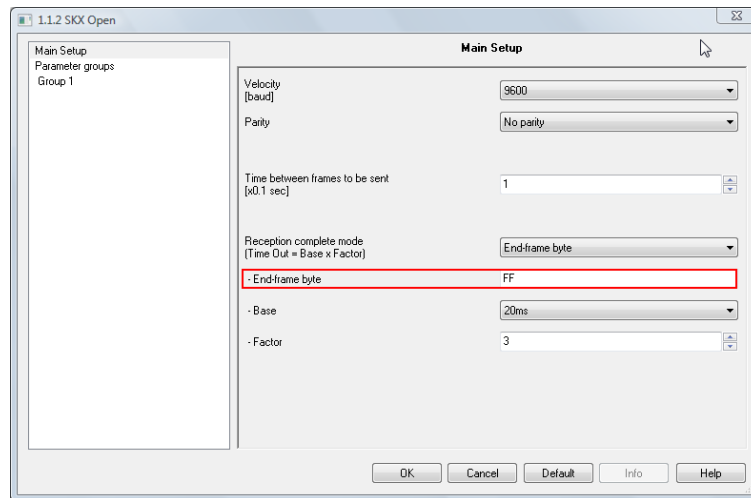


Figure 12. End Frame Byte FFh

#	Time	Service	Flags	Prio	Src.addr	S...	Dest.addr	Destination	Rout	DPT	Type	Data
11...	12:16:55.950	from bus		S	1.1.2		15.15.255	Not Found	6	-	MemoryResponse (S=10)	# bytes: 1 address: B6
11...	12:16:55.979	to bus		S	15.15.255	N...	1.1.2		6	-	ACK (S=10)	
11...	12:16:56.043	to bus		S	15.15.255	N...	1.1.2		6	-	T_Disconnect	
11...	12:17:20.749	from bus		L	1.1.2		0/0/3	Error projector	6	1 byte	Write	\$28 16 %

Figure 13. Error code when End Frame Byte is not received

- Error in Communication Parameters:** If the configured parameters (velocity, parity, etc) in Main Setup tab do not match the ones of the external device an error with code 18 will be sent to the bus.

Example: The external device has a velocity of 9600Bauds and the SKX Open is expected to receive data at 1200 Bauds speed. The error code would be 18 in this case.

- No hexadecimal value:** If a no hexadecimal value is input in a parameter field in SKX Open configuration, this error is sent to the bus with code 01.

Example: The value in the frame parameter field in ETS is 4D5G. In this case, the error code 01 will be sent to the bus during set-up.

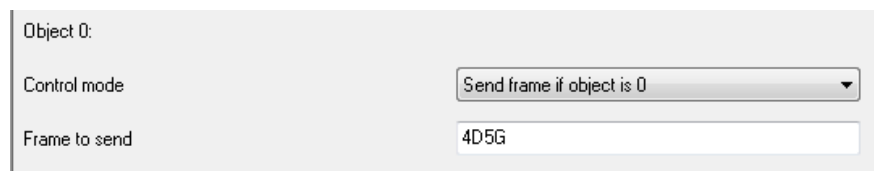


Figure 14. No hexadecimal value

- Odd length value:** If a value with odd length is input in a parameter field in SKX Open configuration, this error is sent to the bus with code 04.

Example: The value in the frame parameter field in ETS is 4D5. In this case, the error code 04 will be sent to the bus during set-up.

Object 0:	
Control mode	Send frame if object is 0
Frame to send	ABcD

Figure 15. Odd length value

- Lower case character:** If a value with a lower case character is input in a parameter field in SKX Open configuration, this error is sent to the bus with code 02.

***Example:** The value in the frame parameter field in ETS is ABcD. In this case, the error code 02 will be sent to the bus during set-up*

Object 0:	
Control mode	Send frame if object is 1
Frame to send	ABcD

Figure 16. Lower case character

- Frame length:** If configured frame length is longer than 10 bytes, the error code 28 will be sent to the bus.

***Example:** The value in the frame parameter field in ETS is 0123456789ABCDEF012345, which is 12bytes long. In this case, the error code 28 will be sent to the bus.*

- The error happens in the current frame:** if when sending a configured frame it contains any of the previous errors, the error code object is sent to the bus, but not the frame.



¡BECOME A USER!

<http://zennioenglish.zendesk.com/portal>

TECHNICAL SUPPORT