Hardware Manual

CAN-GW100/RS232

RS232-CAN Converter

IXXAT
The expert for industrial and automotive communication

IXXAT

Headquarter
IXXAT Automation GmbH
Leibnizstr. 15
D-88250 Weingarten

Tel.: +49 (0)7 51 / 5 61 46-0
Fax: +49 (0)7 51 / 5 61 46-29
Internet: www.ixxat.de
e-Mail: info@ixxat.de

US Sales Office
IXXAT Inc.
120 Bedford Center Road
USA-Bedford, NH 03110

Phone: +1-603-471-0800
Fax: +1-603-471-0880
Internet: www.ixxat.com
e-Mail: sales@ixxat.com

Support

In case of unsolvable problems with this product or other IXXAT products please contact IXXAT in written form by:

Fax: +49 (0)7 51 / 5 61 46-29
e-Mail: support@ixxat.de

For customers from the USA/Canada
Fax: +1-603-471-0880
e-Mail: techsupport@ixxat.com

Copyright

Handbuchnummer: 4.01.0032.20000
Version: 1.4

1 Lead-in 7

1.1 Overview 7

2 Connections 8

2.1 Connector allocation 8

2.1.1 Power supply (X1) 8

2.1.2 Serial interface RS232 (X2) 9

2.1.3 CAN (X3) 9

2.2 Ground connections 9

3 Description of functions 10

3.1 Introduction 10

3.2 RS232-CAN gateway 10

3.3 RS232-CANopen gateway 11

3.4 Data structure of the configuration files 14

3.4.1 General settings [General] 16

3.4.1.1 Product name (ProductName) 16

3.4.1.2 Version number (TemplateVersion) 16

3.4.1.3 Operation mode (OperationMode) 16

3.4.1.4 Timeout 16

3.4.2 User settings [User] 16

3.4.2.1 Configuration name (ConfigAlias) 16

3.4.3 RS232 settings [RS232] 16

3.4.3.1 Baudrate (RS232baudrate) 16

3.4.3.2 Number of databits (Databits) 16

3.4.3.3 Parity (Parity) 17

3.4.3.4 Flow control (Handshake) 17

3.4.4 CAN-GW100/RS232 settings [CANmode] 17

3.4.4.1 Baudrate (CANbaudrate) 17

3.4.4.2 Frame format (FrameFormat) 17

3.4.4.3 Send identifier (SendID) 17

3.4.4.4 Receive identifier (ReceiveID) 18

3.4.5 CAN-GW100/RS232 settings [COPmode] 18

3.4.5.1 Baudrate (CANopenBaudrate) 18

3.4.5.2 CANopen node number (CANopenNode) 18

3.4.5.3 Heartbeat time (HBTime) 18

3.4.5.4 Receive PDO (RxPDO) 18

3.4.5.5 Receive PDO type (RxPDOtype) 18
3.4.5.6 Transmit-PDO (TxPDO) 19
3.4.5.7 Transmit-PDO type (TxPDOtype) 19
3.4.5.8 Byte stream flow control (ByteStreamExtension) 19
3.5 Default configuration 20
4 Download tool 22
4.1 Configuration with Windows console program 22
4.1.1 Creating a configuration file 23
4.1.2 Download of a configuration 23
4.1.3 Displaying the current configuration 24
4.1.4 Saving the current configuration 25
5 Configuration tool 26
5.1 Configuration with Windows application 26
5.1.1 Default configuration 26
5.1.2 Loading and saving a configuration 27
5.1.3 Setting up a connection 27
5.1.4 Reading the current configuration 29
5.1.5 Downloading a configuration 30
5.1.6 Disconnecting 30
5.2 Configuration cable 30
6 Displays 31
6.1 Normal mode 31
6.1.1 Power LED 31
6.1.2 CAN LED 31
6.1.3 Status LED 31
6.1.4 RS232 LED 31
6.2 Configuration mode 32
6.3 Error state 32
7 Notes on EMC 33
7.1 Shield concept 33
8 Appendix 34
8.1 Support 34
8.2 Returning hardware 34
8.3 Technical specifications 35

8.4 Sources of data sheets 35
8.5 EC conformity declaration 36

1 Lead-in

1.1 Overview

Congratulations on your purchase of the IXXAT CAN-GW100/RS232, a high-quality electronic component developed and manufactured according to the latest technological standards.

This manual is intended to familiarize you with your CAN-GW100/RS232. Please read this manual before initial startup.

The CAN-GW100/RS232 enables devices with only one serial port a simple, configurable access to CAN and CANopen networks. CAN-GW100/RS232 provides two operation modes for this purpose.

In the CAN operation mode (CANmode), received CAN data are transmitted transparently to the RS232 interface. Data received via RS232 are mapped into CAN telegrams and transmitted. One configurable identifier each is available for transmission and reception.

In the CANopen operation mode (COPmode), the CAN-GW100/RS232 operates as a CANopen node, where the serial data are stored as a byte stream object in the manufacturer-specific object dictionary range.
The supported CANopen features are:

- 1 server SDO expedited, non-expedited, no CRC check
- 1 TX PDO static mapping
- 1 RX PDO static mapping
- Emergency message
- Heartbeat producer
- NMT slave

The communication interfaces and operation modes are configured by means of a configuration file, which is saved on the device via a loading program.

2 Connections

2.1 Connector allocation

The pin allocation for the Industrial DIN rail version is shown in Fig. 2-1.

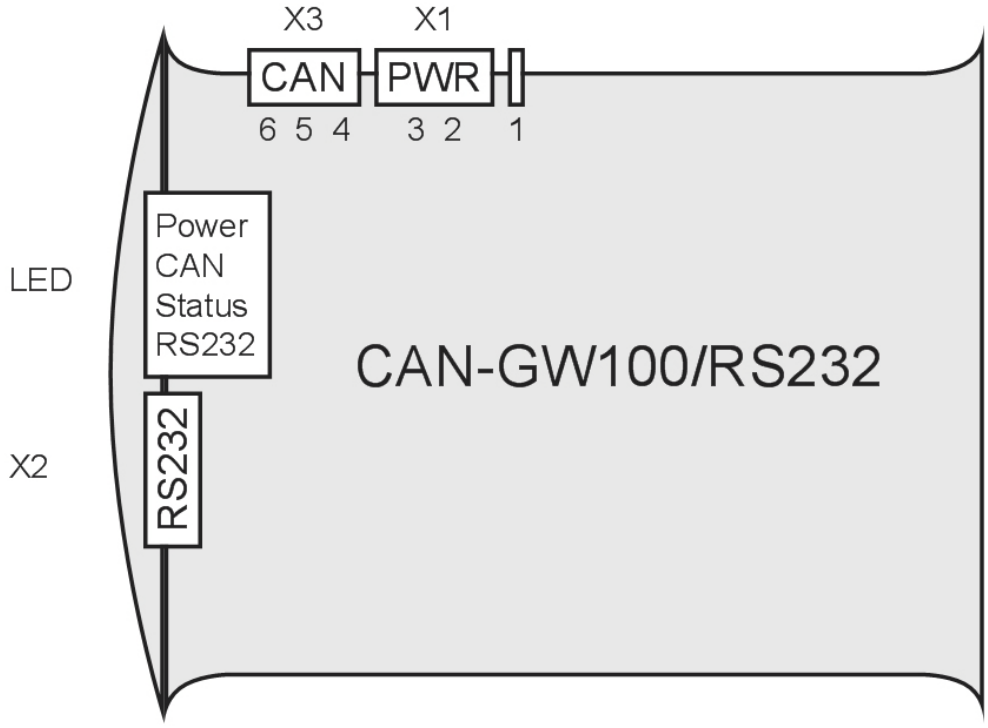


Fig. 2-1: Connector arrangement

2.1.1 Power supply (X1)

The device is supplied with DC voltage of 9 V - 36 V. The pin allocation is shown in Table 2-1. The CAN-GW100/RS232 is protected against polarity reversal, under-voltage and over-voltage.

In the event of polarity reversal or under-voltage, it is switched off, with over-voltage an internal fuse is triggered.

Terminal	Signal
1	PE
2	PWR (+)
3	GND (-)

Table 2-1: Terminal allocation Power

2.1.2 Serial interface RS232 (X2)

The signals of the serial port are connected to the 9-pin Sub-D connector X2 (see Table 2-2).

Pin no. X2	Signal
1	DCD
2	RX
3	TX
4	DTR
5	GND
6	DSR
7	RTS
8	CTS
9	RI

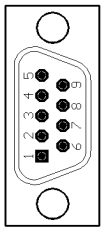


Table 2-2: Pin allocation RS232

2.1.3 CAN (X3)

CAN is available on X3 with a bus interface according to ISO 11898-2 (see Table 2-3).

Terminal	Signal
4	CAN High
5	CAN Low
6	GND

Table 2-3: Terminal allocation CAN1

2.2 Ground connections

In the galvanically isolated version, the GND of CAN (X3) is isolated from the rest of the circuit, the GND of the serial port (X2) is connected to the GND of the power supply (X1).

In the version without galvanic isolation, all GND connections (X1, X2, X3) are connected with each other.

The shield connection of the serial port (X2) and the PE connection (X1 – terminal 1) are connected with each other both in the version with and in the version without galvanic isolation.

3 Description of functions

3.1 Introduction

The CAN-GW100/RS232 offers the two operation modes CANmode (RS232-CAN Gateway) and COPmode (RS232-CANopen Gateway).
In CANmode, the device transmits the data that arrives on the serial port to the CAN bus under a configurable identifier. In the same way, data that are received via CAN by means of another configurable identifier are passed on to the serial port.

The identifiers, CAN parameters and the parameters of the serial port used can be freely configured.

In COPmode, the databytes that arrive on the serial port are transmitted on the CAN bus via TxPDO. Conversely, the data received via a certain RxPDO are passed on to the serial port. On the CANopen side, the **"Byte Stream Protocol"** is used as the communication protocol.
The CAN-GW100/RS232 can be configured via the serial port with the aid of a download tool.

3.2 RS232-CAN gateway

Table 3-1 shows the functions and the associated key words, relevant for the CANmode.

Function	Possible settings	Key words
Operation mode	CANmode	OperationMode (under [General])
Timeout	0 - 6,5 s	Timeout (under [General])
RS232 parameter	1. Baudrate 2. Number of databits 3. Parity 4. Flow control	RS232baudrate Databits Parity Handshake (all under [RS232])
Baudrate for CAN bus	1. Setting of a CiA baudrate 2. Setting the bus timing register BTR0 / BTR1	CANbaudrate (under [CANBus])
Frame Format for CAN bus	1. Standard Frame Format (11bit identifier) 2. Extended Frame Format (29 bit identifier)	FrameFormat (under [CANBus])

CAN identifier for transmission of the data to the CAN bus	All CAN identifiers are possible	SendID (under [CANmode])
CAN identifier for receiving data from CAN bus	All CAN identifiers are possible	ReceiveID (under [CANmode])

Table 3-1

3.3 RS232-CANopen gateway

Table 3-2 shows the functions and the associated key words, relevant for the COPmode.

Function	**Possible settings**	**Key words**
Operation mode	COPmode	OperationMode (under [General])
Timeout	0 - 6,5 s	Timeout (under [General])
RS232 parameter	1. Baudrate 2. Number of databits 3. Parity 4. Flow control	RS232baudrate Databits Parity Handshake (all under [RS232])
CANopen baudrate	CiA baudrates are possible	CANopenBaudrate (under [COPmode])
CANopen node number	1-127	CANopnNode (under [COPmode])
Heart Beat Time	0- 32 s	HBTime (under [COPmode])
Receive PDO	PDO ID – all free CANopen Ids PDO Type – 0..240, 252..255	RxPDO RxPDOtype (under [COPmode])
Transmit PDO	PDO ID – all free CANopen Ids PDO Type – 0..240, 252..255	TxPDO TxPDOtype (under [COPmode])
Flow control on CAN	On / off	ByteStreamExtension (under [COPmode])

Table 3-2

In COPmode, the CAN-GW100/RS232 is an CANopen slave. Nearly all parameters may be configured over the object dictionary using SDO.

Table 3-3 shows the Object Dictionary of the CAN-GW100/RS232 in the RS232-CANopen gateway mode.

Idx	*Sidx*	*Name (Reference)*	*At-trib*	*Ma-pable*	*Obj Type*	*DataType*	*Default Value, Range*
1000	00	device type	ro	n	VAR	Unsigned32	0x00000000
1001	00	error register	ro	y	VAR	Unsigned8	0x00
1003		pre-defined error field			ARRAY	Unsigned32	
	00	number of entries	rw	n		Unsigned8	0x00
	01	standard error field	ro	n		Unsigned32	0x00
	02	standard error field	ro	n		Unsigned32	0x00
	03	standard error field	ro	n		Unsigned32	0x00
	04	standard error field	ro	n		Unsigned32	0x00
1005	00	COB-ID SYNC message	rw	n	VAR	Unsigned32	0x00000080
1008	00	manufacturer device name	ro	n	VAR	Vis-String	"IXXAT COP-link"
1009	00	manufacturer hardware version	ro	n	VAR	Vis-String	"01.01"
100A	00	manufacturer software version	ro	n	VAR	Vis-String	"1.1.2.0"
100C	00	Guard time	rw	n	VAR	Unsigned16	0
100D	00	Life time factor	rw	n	VAR	Unsigned8	0
1010		store parameters			ARRAY	Unsigned32	
	00	largest Sidx supported	ro	n	ARRAY	Unsigned8	0x04
	01	save all parameters	rw	n	VAR	Unsigned32	
1011		restore parameters			ARRAY	Unsigned32	
	00	largest Sidx supported	ro	n		Unsigned8	0x04
	01	restore all default para-meters	rw	n		Unsigned32	0x00
1014	00	COB-ID Emergency	rw	n	VAR	Unsigned32	0x80+Node-ID
1016		Consumer Heartbeat Time			ARRAY	Unsigned32	
	00	Number entries	ro	n		Unsigned8	0x05
	01	Consumer Heartbeat Time	rw	n		Unsigned32	0x00
	02	Consumer Heartbeat Time	rw	n	VAR	Unsigned32	
	03	Consumer Heartbeat Time	rw	n	VAR	Unsigned32	
1017	00	producer heartbeat time	rw	n	VAR	Unsigned16	0x00
1018		identity object	ro		RECORD	Identity	
	00	number of entries	ro	n		Unsigned8	0x04
	01	Vendor-ID	ro	n		Unsigned32	0x04
	02	Product code	ro	n		Unsigned32	341
	03	Revision number	ro	n		Unsigned32	0x00

Idx	*Sidx*	*Name (Reference)*	*At-trib*	*Ma-pable*	*Obj Type*	*DataType*	*Default Value, Range*
	04	Serial number	ro	n		Unsigned32	
1200		server SDO 1			RECORD	SDO Param	
	00	number of entries	ro	n		Unsigned8	0x02
	01	COB-ID client->server	ro	n		Unsigned32	600 + Node-ID
	02	COB-ID server->client	ro	n		Unsigned32	580 + Node-ID
1400		receive PDO 1 comm. parameter			RECORD	PDO Comm-Par	
	00	number of entries	ro	n		Unsigned8	0x02
	01	COB-ID used by PDO	rw	n		Unsigned32	200h + Node-ID
	02	transmission type	rw	n		Unsigned8	0xff
1600		receive PDO 1 mapping parameter			RECORD	PDO Mapping	
	00	number of entries	ro	n		Unsigned8	0x01
	01	1st object	ro	n		Unsigned32	0x21000140
1800		transmit PDO 1 comm. parameter			RECORD	PDO Comm-Par	
	00	number of entries	ro	n		Unsigned8	0x05
	01	COB-ID used by PDO	rw	n		Unsigned32	180h + Node-ID
	02	transmission type	rw	n		Unsigned8	0xff
	05	event timer	rw	n		Unsigned16	0x00
1A00		transmit PDO 1 mapping parameter			RECORD	PDO Mapping	
	00	number of entries	ro	n		Unsigned8	0x01
	01	1st object	ro	n		Unsigned32	0x20000140

2000		Byte Stream TxData			RECORD		
	00	Number of elements	ro	n		Unsigned8	0x01
	01	1st BSC transmit channel	ro	y		Unsigned64	
2001	00	Timeout Time	rw	n	VAR	Unsigned16	
2002		CANopen parameters			RECORD		
	00	number of elements	ro	n		Unsigned8	3
	01	Using manugacturer specific Byte stream protocol extension	rw	n		Unsigned8	1

Idx	*Sidx*	*Name (Reference)*	*At-trib*	*Ma-pable*	*Obj Type*	*DataType*	*Default Value, Range*
	02	Low Water Mark	rw	n		Unsigned8	25 {0..100}
	03	High Water Mark	rw	n		Unsigned8	65 {0..100}
2003		RS232 parameters			RECORD		
	00	number of elements	ro	n		Unsigned8	4
	01	RS232 baudrate	rw	n		Unsigned8	4 0 – 600 1 – 1200 2 – 2400 3 – 4800 4 - 9600 5 - 19200 6 - 38400 7 – 57600 8 – 115200
	02	RS232 data bit number	rw	n		Unsigned8	8 {7,8}
	03	RS232 parity	rw	n		Unsigned8	0 0 – oo 1 – odd 2 - even
	04	RS232 handshake	rw	n		Unsigned32	0 0 – no 1 – software 2 - hardware
2100	00	Byte Stream RxData			RECORD		
	00	Number of elements	ro	n		Unsigned8	0x01
	01	1st BSC receive channel	wo	y		Unsigned64	

Table 3-3

3.4 Data structure of the configuration files

A template of the ASCII configuration file can be generated with the download tool (see section 5). The required configuration can be created easily with the template and a text editor. The data structure is based on the Windows INI format.

Possible settings of the configuration file are listed in the following:

```
[General]
 ProductName = CAN-GW100/RS232    ; !Do not change this!
 TemplateVersion = 1.0            ; !Do not change this!
 OperationMode = COPmode          ; CANmode and COPmode are possible
 Timeout = 100                    ; 0-65000 (0-6500 ms).

[User]
 ConfigAlias = "IXXAT default"

[RS232] ;  Parameters of serial communication
 RS232baudrate = 9600
 Databits = 8                     ; 7,8
 Parity = no                      ; no, odd, even
 Handshake = no                   ; no, software, hardware

[CANmode]
 CANbaudrate = 100
 FrameFormat = std                ; std (11 bit ID) or ext (29 bit ID)
 ReceiveID = 0x100
 SendID  = 0x101

[COPmode]
 CANopenBaudrate = 4
 CANopenNode = 0x1
 HBTime  = 10000                  ; if 0 then Heartbeat Producer is disabled
 RxPDO  = 0x201
 RxPDOtype = 0xFF
 TxPDO = 0x181
 TxPDOtype = 0xFF
 ByteStreamExtension = yes        ; IXXAT Byte Stream Protocol extension
```

3.4.1 General settings [General]

There are four parameters in the ASCII file under the key word [General]:

3.4.1.1 Product name (ProductName)

The product name states for which product the configuration file is intended. The name is used for a compatibility check and must not be changed.

3.4.1.2 Version number (TemplateVersion)

The version number of the ASCII file structure. It is used for a compatibility check and likewise must not be changed.

3.4.1.3 Operation mode (OperationMode)

Via this parameter the user defines whether the CAN-GW100/RS232 functions as an "RS232 – CAN Gateway" or as an "RS232 – CANopen Gateway". The possible values are CANmode and COPmode. (e.g. OperationMode = COPmode)

3.4.1.4 Timeout

The data of the serial port are transmitted on the CAN bus in blocks of 8 databytes each. If fewer than 8 databytes, but at least 1 databyte, are received via the serial port, and no new databyte is received until expiry of the time set under the Timeout parameter, the data received up to this time are transmitted in a smaller block via the CAN bus. The resolution is 0.1 ms (e.g. Timeout = 100 corresponds to 10 ms)

3.4.2 User settings [User]

3.4.2.1 Configuration name (ConfigAlias)

The user can set a character chain of up to 15 characters for the designation of the individual configuration. The identification key is stored in the device and can be read out with the download tool.

3.4.3 RS232 settings [RS232]

3.4.3.1 Baudrate (RS232baudrate)

The following baudrates can be set for the RS232 interface:

600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

(e.g. RS232baudrate = 9600)

3.4.3.2 Number of databits (Databits)

This parameter defines whether the data are transmitted with a length of 7 or 8 bits. (e.g. databits = 8)

3.4.3.3 Parity (Parity)

Parity defines whether a parity bit is to be transmitted at the same time. The possibilities are even parity (even), odd parity (odd) or no parity (no) (e.g. Parity = no).

3.4.3.4 Flow control (Handshake)

The handshake parameter defines the type of flow control of the serial data transmission. Possible settings are no handshake (no), software handshake (software) or hardware handshake (hardware) (e.g. Handshake = no).

3.4.4 CAN-GW100/RS232 settings [CANmode]

3.4.4.1 Baudrate (CANbaudrate)

There are two possibilities for setting the baudrate:

1. **Setting a CiA baudrate:**

   10 kbit/s, 20 kbit/s, 50 kbit/s, 100 kbit/s, 125 kbit/s, 250 kbit/s, 800 kbit/s and 1000 kbit/s can be set (e.g. CANbaudrate=1000).

2. **Setting via bit timing register**

   The baudrate can be set via the bit timing registers BTR0 and BTR1 of the controller. Baudrates can thus also be selected that are not listed in the table.

   The values for the bit timing registers BTR0 and BTR1 are determined according to the specifications for the Philips CAN controller SJA1000 with 16 MHz clock frequency (see data sheet SJA1000).

   However, prescaler PSC (bits 0..5 of the BTR0) must not be larger than 0x1F, with the exception of baudrate 10 kbit, which can be set via 0x31/0x1C.

   (e.g. CAN baudrate = 0x04/0x14 for 200 kbit/s).

3.4.4.2 Frame format (FrameFormat)

The frame format states in which format the messages are transmitted on the CAN bus. Standard frames (11 bit identifiers) or extended frames (29 bit identifiers) can be selected.
(e.g. FrameFormat = std)

3.4.4.3 Send identifier (SendID)

The data received by the serial port are passed on via the CAN bus. The identifier used for data transmission on CAN is defined with the parameter SendID. (e.g. SendID=0x100)

3.4.4.4 Receive identifier (ReceiveID)

CAN messages with the identifier given under ReceiveID are received by the CAN-GW100/RS232, the data are passed on to the serial port. (e.g. Receive-ID=0x200)

3.4.5 CAN-GW100/RS232 settings [COPmode]

3.4.5.1 Baudrate (CANopenBaudrate)

The baudrate is defined via the CiA baudrates table:

8 - 10kbit/s;
7 - 20kbit/s;
6 - 50kbit/s;
5 - 100kbit/s;
4 - 125kbit/s;
3 - 250kbit/s;
2 - 500kbit/s;
1 - 800kbit/s;
0 – 1000kbit/s.

(e.g. CANopenBaudrate = 4)

3.4.5.2 CANopen node number (CANopenNode)

This parameter defines the CANopen node number of the device.
(e.g. CANopenNode = 0x04)

3.4.5.3 Heartbeat time (HBTime)

This parameter is used to define the time interval in ms between the heartbeat messages that are transmitted by the heartbeat producer of the device. With HBTime = 0 the heartbeat is switched off.

(e.g. HBTime = 10000 means 10 seconds)

3.4.5.4 Receive PDO (RxPDO)

Message identifier of the receive PDO. The device defines static mapping of the byte stream objects in the receive PDO. CAN messages with the identifier given under RxPDO are received by the CAN-GW100/RS232 as RxPDO of the byte stream protocol, the data are passed on to the serial port. (e.g. RxPDO = 0x201)

3.4.5.5 Receive PDO type (RxPDOtype)

The parameter RxPDOtype defines how the data of the received PDO are processed further: event-controlled, cyclically or synchronously. If a synchronous PDO is set (0..240,252), the sync message is interpreted as a timeout event (e.g. RxPDOtype = 0xFF).

3.4.5.6 Transmit-PDO (TxPDO)

Data arriving on the serial port are transmitted by the CAN-GW100/RS232 on the CAN bus with the identifier given under TxPDO as a TxPDO of the byte stream protocol (e.g. TxPDO = 0x181)

3.4.5.7 Transmit-PDO type (TxPDOtype)

The parameter TxPDOtype defines how the PDO is sent. The serial communication via RS232 is generally asynchronous, i.e. as soon as sufficient data have been received via the serial port, the PDO is sent. If a synchronous PDO type is set (0..240,252), the sync message is interpreted as a timeout event (e.g. TxPDOtype = 0xFF)

3.4.5.8 Byte stream flow control (ByteStreamExtension)

The CAN-GW100/RS232 supports IXXAT-specific byte stream commands, which are used for data flow control via CANopen. This increases the stability of the data transfer (e.g. ByteStreamExtension = yes)

3.5 Default configuration

The CAN-GW100/RS232 is supplied with the following standard configuration:

```
; this is a configuration file for IXXAT the CAN-GW100/RS232

[General]
 ProductName = CAN-GW100/RS232      ; !Do not change this!
 TemplateVersion = 1.0              ; !Do not change this!
 OperationMode = CANmode            ; CANmode and COPmode are possible
                                    ; Timeout defines time CAN-GW100/RS232 waits for
                                      new data at the RS232
                                    ; without sending incomplete data block via CAN
 Timeout = 100                      ; 0-65000 (0-6500 ms).

[User]
 ConfigAlias = "IXXAT default"

[RS232]   ; Parameters of serial communication
 RS232baudrate = 9600               ; 600, 1200, 2400, 4800, 9600, 19200, 38400,
                                      57600, 115200 bit/s
 Databits = 8                       ; 7,8
 Parity = no                        ; no, odd, even
 Handshake = no                     ; no, software, hardware

[CANmode]
 ; CAN bus baudrate can be defined via CiA value or alternatively
 ; per registers BTR0 and BTR1 as BTR0/BTR1 (see table below)
 ; Baud Rate   CiA value     BTR0/BTR1
 ;    1 Mb/s       1000      0x00/0x14
 ;  800 kb/s        800      0x00/0x16
 ;  500 kb/s        500      0x00/0x1C
 ;  250 kb/s        250      0x01/0x1C
 ;  125 kb/s        125      0x03/0x1C
 ;  100 kb/s        100      0x04/0x1C
 ;   50 kb/s         50      0x09/0x1C
 ;   20 kb/s         20      0x18/0x1C
 ;   10 kb/s         20      0x31/0x1C
 CANbaudrate = 100
 FrameFormat = std   ; std (11 bit ID) or ext (29 bit ID)
 ReceiveID   = 0x100
 SendID      = 0x101
```

[COPmode]

```
; Table of CiA baudrates
;  #   Baudrate, kbps
;  0   1000
;  1   800
;  2   500
;  3   250
;  4   125
;  5   reserved (100)
;  6   50
;  7   20
;  8   10
 CANopenBaudrate  = 4
 CANopenNode  = 0x1
 HBTime  = 10000                ; if 0 then heartbeat producer is disabled
 RxPDO = 0x201
 RxPDOtype = 0xFF
 TxPDO = 0x181
 TxPDOtype = 0xFF
 ByteStreamExtension = yes      ; IXXAT byte stream protocol extension
```

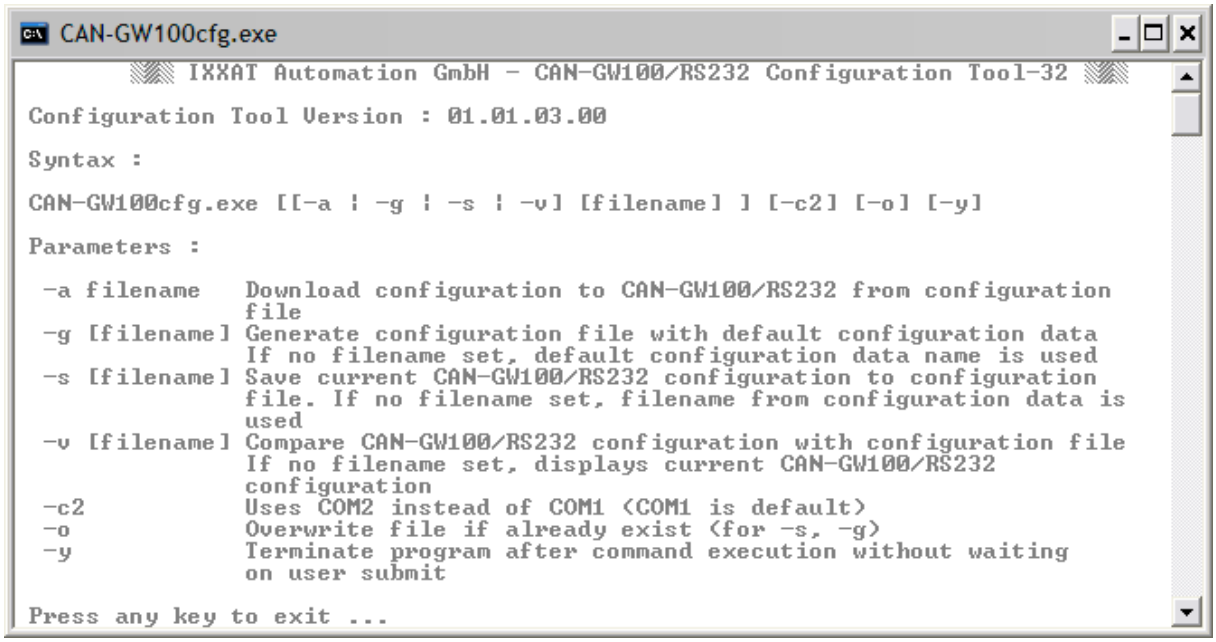
4 Download tool

The Windows console program CAN-GW100cfg.exe and alternatively the Windows application CANL2cfgGUI.exe are available for configuration of the CAN-GW100/RS232.

4.1 Configuration with Windows console program

The program is operated via call parameters. The following possibilities are available to run the program

- In Windows you can change to the path of the program CAN-GW100cfg.exe via "Start – Run – Browse". Select by clicking and add the required parameters.
- You can open a DOS box under "Start – Programs – MS-DOS-Prompt", change to the directory containing the program CAN-GW100cfg.exe and then run it with the corresponding parameters.
- You can create a batch file or a link.



4.1.1 Creating a configuration file

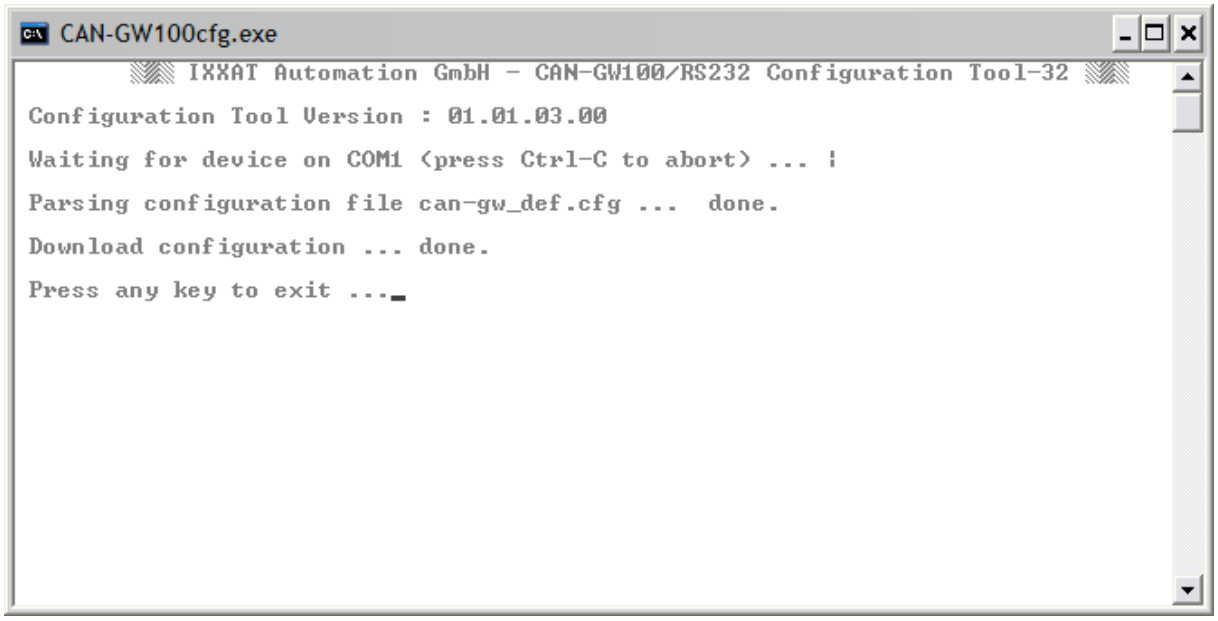
The following steps are necessary to create a new configuration:

- Call of the program CAN-GW100cfg.exe with the parameter –g (CAN-GW100cfg -g). This creates the template file can-gw100_rs232_def.cfg. By entering a file name after the parameter –g (e.g. CAN-GW100cfg –g myconfig.cfg), the template file is created with this file name.
- The required configuration can now be created by editing the template file with a text editor.

4.1.2 Download of a configuration

To save a configuration, the serial port of the switched off CAN-GW100/RS232 and the serial port of the PC must be connected with the cable supplied. Then the configuration program CAN-GW100cfg.exe is called with the parameter –a <Filename> [interface] (e.g. CAN-GW100cfg –a myconfig.cfg). When the message "Waiting for device" appears on the screen, the device can be supplied with power, the device goes into configuration mode (see section 5.2) and the download is performed.

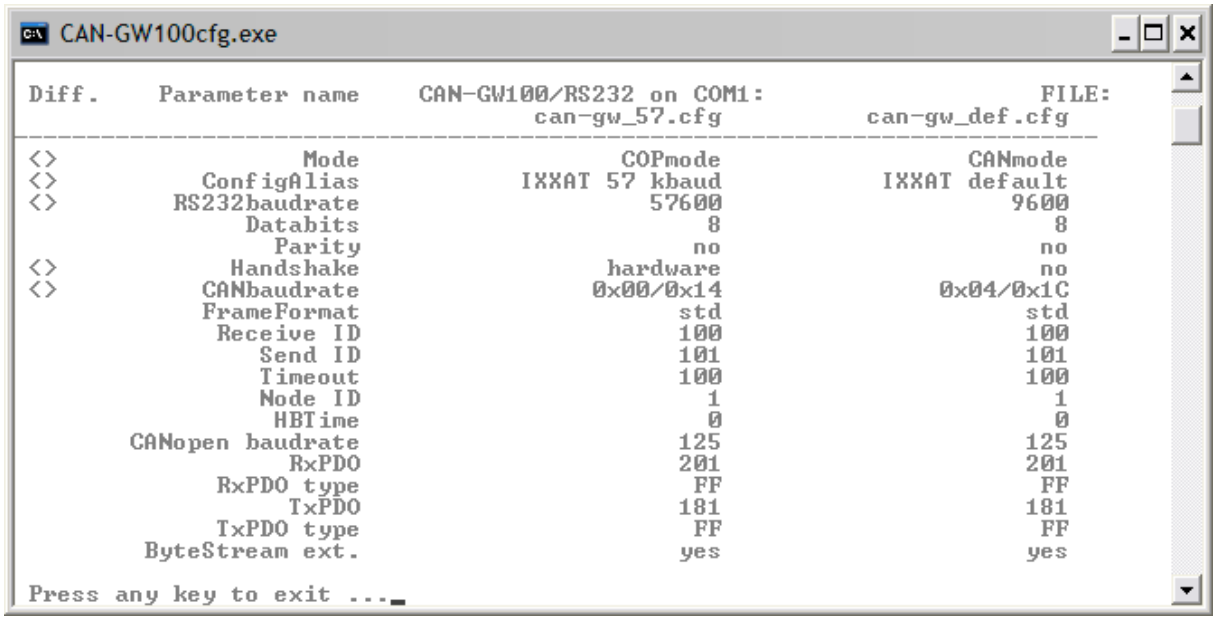
In addition to the configuration settings, the file name of the configuration file is also saved in the CAN-GW100/RS232 (max. 15 characters).



4.1.3 Displaying the current configuration

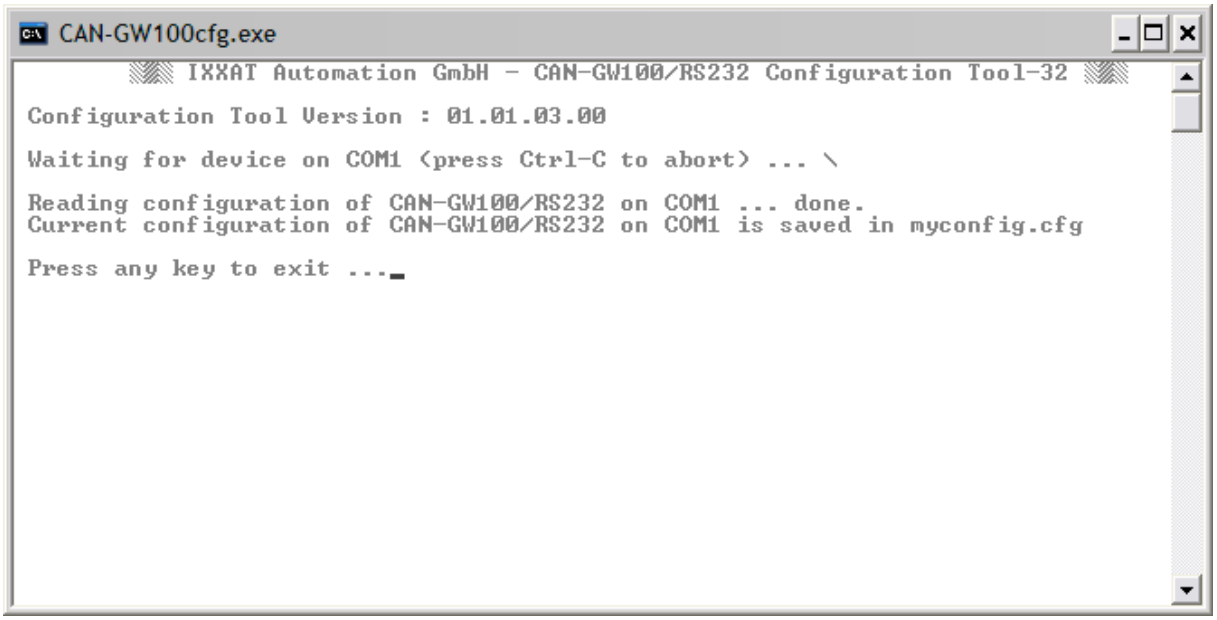
The current configuration can be read out with the parameter –v (e.g. CAN-GW100cfg –v). After the message "Waiting for CAN-GW100/RS232 " appears on the screen, the device must be switched off briefly and then switched on again to enter the configuration mode. The file name of the configuration file and the current settings are displayed.

If a file name is entered after the parameter –v, the configuration of the device is compared with the parameters of the stated file (e.g. CAN-GW100cfg –v device.cfg). Differences are displayed with the symbol <>.



4.1.4 Saving the current configuration

The current configuration of the device can be saved in a file with the parameter –s (e.g. CAN-GW100cfg –s –myconfig.cfg). If no file name is entered, the program uses the configuration file name stored in the CAN-GW100/RS232. After the message "Waiting for CAN-GW100/RS232 " appears on the screen, the device must be switched off briefly and then switched on again to enter the configuration mode.



5 Configuration tool

5.1 Configuration with Windows application

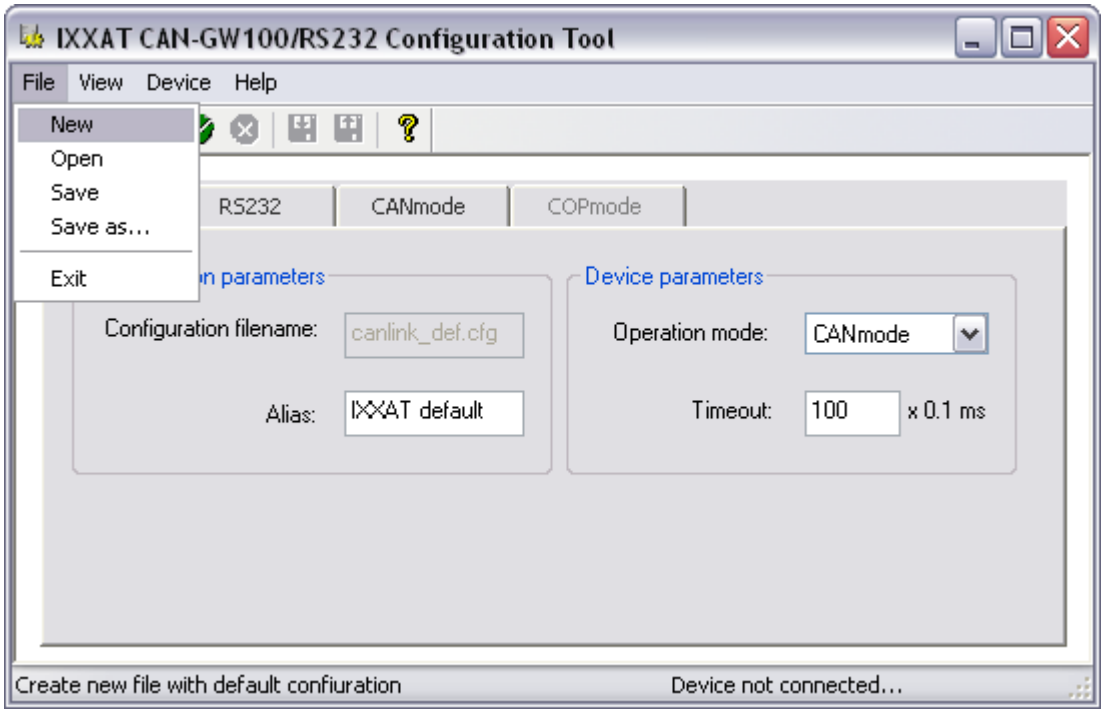
The Windows application CAN-GW100cfgGUI.exe is available as an alternative to the Windows console program. The program allows the user to configure the device by means of a graphic interface.

The Windows application displays the configuration file on four configuration pages. Each page lists parameters of the relevant range.

- **"General"** – configuration name, active operation mode, timeout.
- **"RS232"** – parameters for serial port (baudrate, data flow control etc.).
- **"CANmode"** – parameters for CAN-on-RS232 operation mode (TxID, RxID, baudrate etc.).
- **"COPmode"** – parameters for CANopen-on-RS232 operation mode (TxPDO, RxPDO, node number etc.).

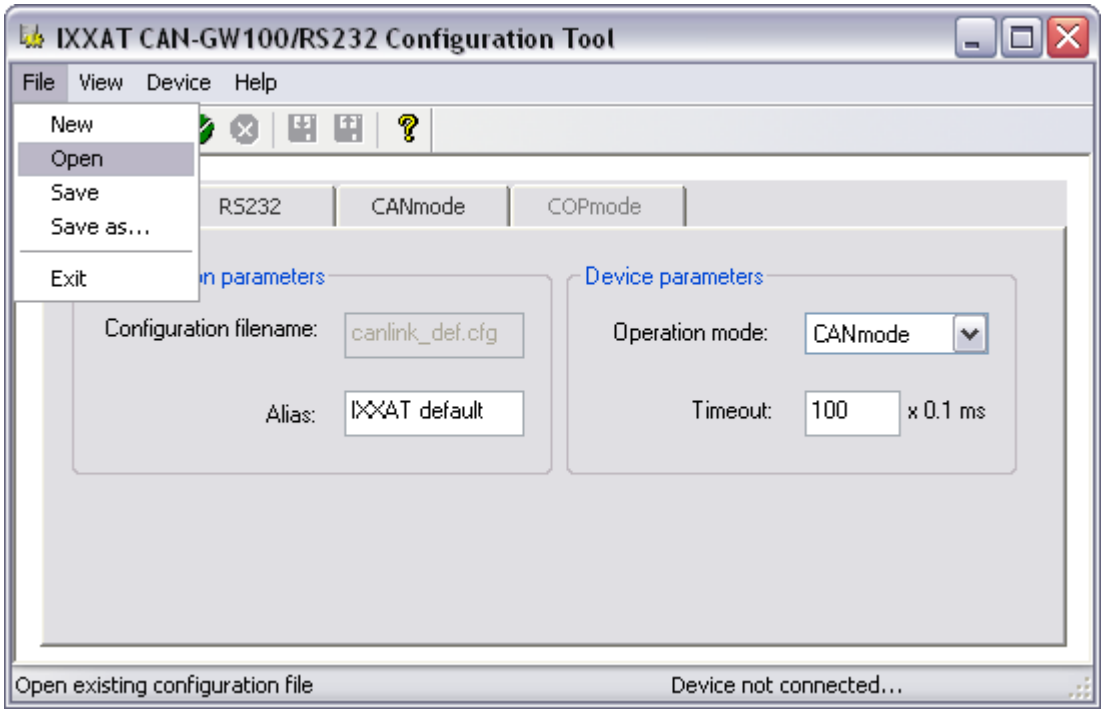
5.1.1 Default configuration

The original configuration of the CAN-GW100/RS232 can be recovered via the menu *"File -> New"*. This can then be written in the device or in a file.



5.1.2 Loading and saving a configuration

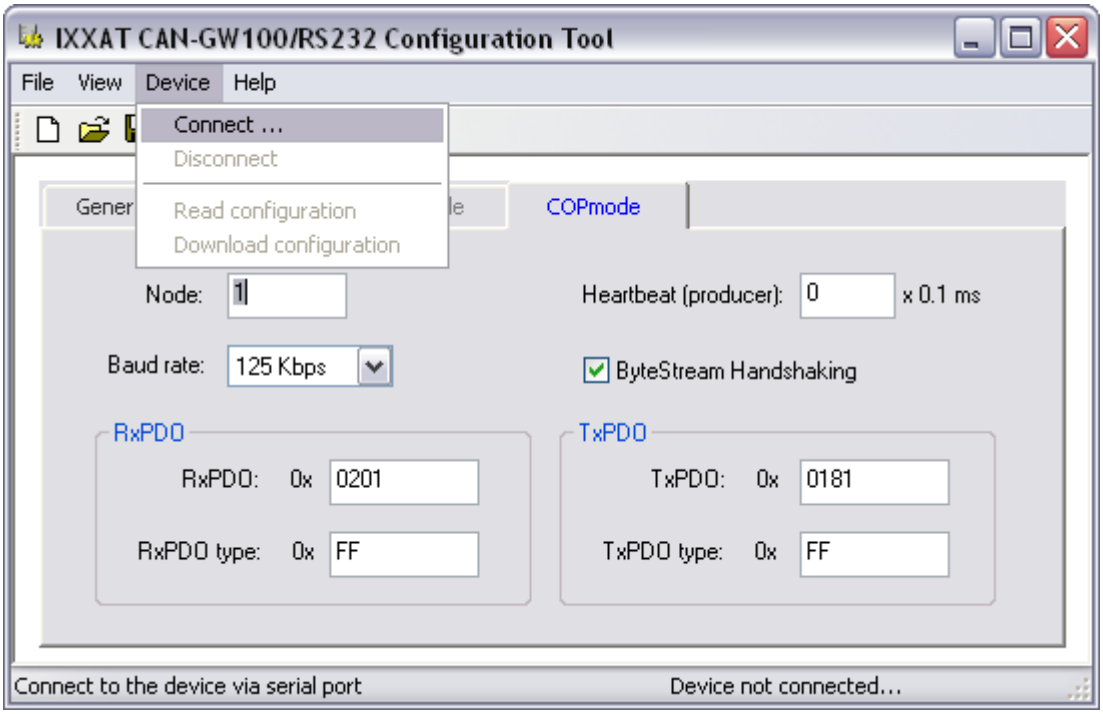
The configuration is loaded from a file via *"File -> Open"*.



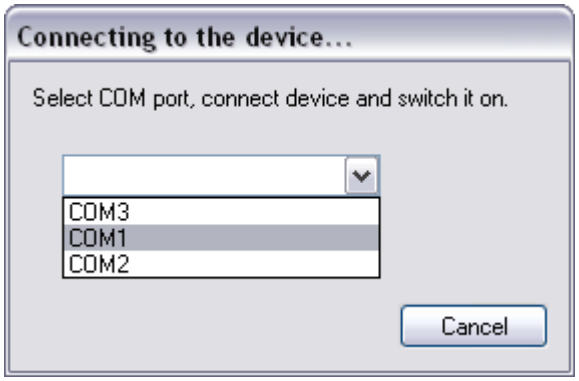
The current configuration can be edited and saved in the same file with *"File -> Save"*. If the configuration should be saved in a different file, use *"File -> Save as..."*.

5.1.3 Setting up a connection

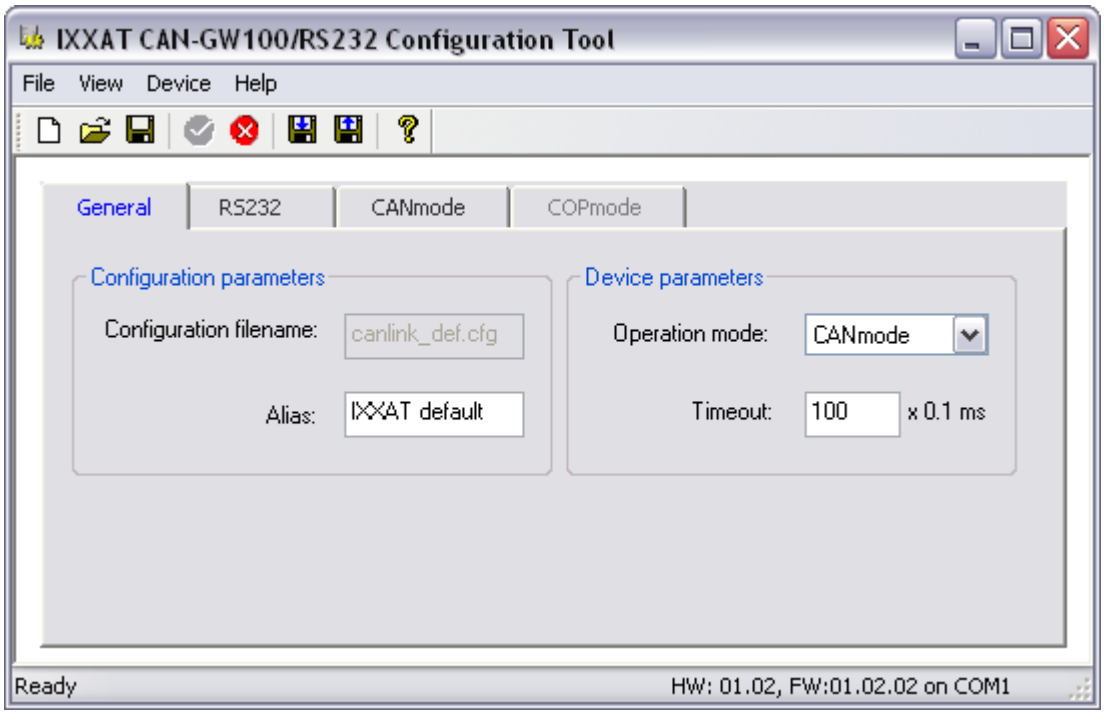
In order to configure the device, the serial port of the switched off CAN-GW100/RS232 and the serial port of the PC must be connected with the cable supplied. Please select *"Device -> Connect ..."*.



Then you are asked to enter the serial port to which you have connected the device.

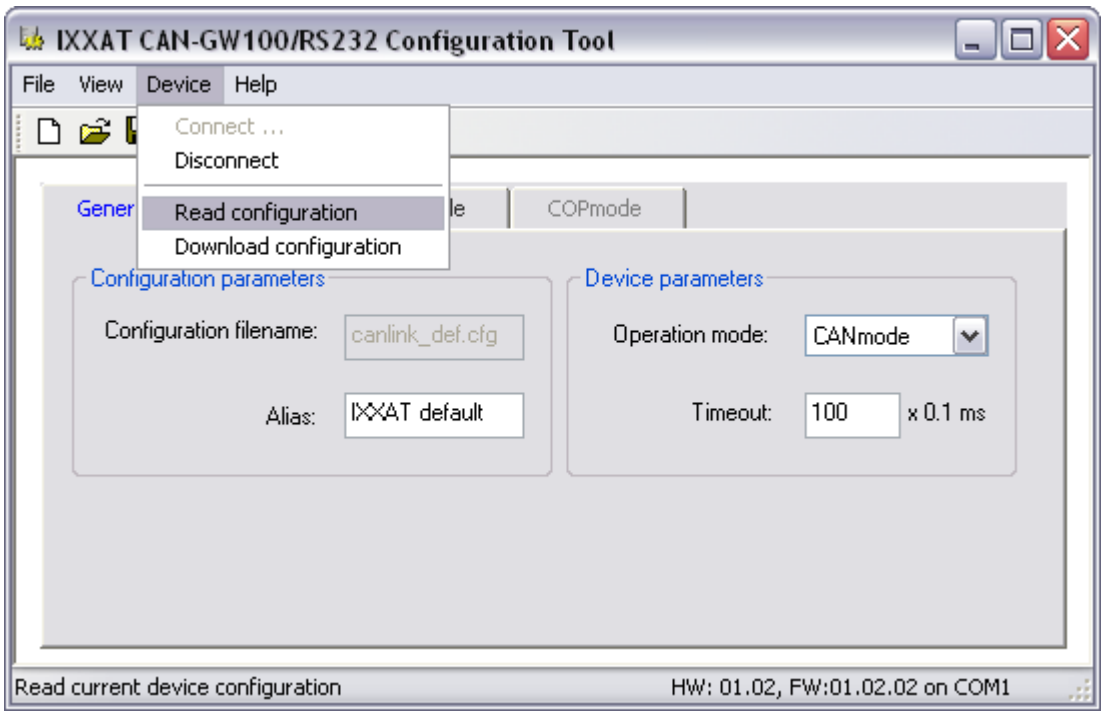


After selecting the serial port, the device can be supplied with power. As soon as there is voltage on the CAN-GW100/RS232, the device is detected and checked for compatibility. The hardware version and firmware version are displayed in the status bar.



5.1.4 Reading the current configuration

The current configuration of the device can be read out via *"Device -> Read configuration"*.



5.1.5 Downloading a configuration

To save a configuration in the device, please select *"Device -> Download configuration"*.

5.1.6 Disconnecting

The connection between the device and the configuration tool can be interrupted at any time via *"Device -> Disconnect"*.

5.2 Configuration cable

In order to enter the configuration mode, a connection cable with Sub-D9 sockets at both ends is required for the serial port. The connection of the two Sub-D9 sockets is listed in table 5-1. The connector allocation corresponds to that of a serial laplink cable. This cable is also suitable for normal operation mode.

Sub-D9 socket 1	Pin	Pin	Sub-D9 socket 2
DCD	1	4	DTR
RX	2	3	TX
TX	3	2	RX
DTR	4	1	DCD
GND	5	5	GND
DSR	6	-	
RTS	7	8	CTS
CTS	8	7	RTS
RI	9	9	RI

Table 5-1: Connection list of the configuration cable

6 Displays

The CAN-GW100/RS232 has four two-color LEDs (see Fig. 2-1). The LEDs behave as follows depending on the operation mode of the CAN-GW100/RS232.

6.1 Normal mode

6.1.1 Power LED

The Power LED is lit when the CAN-GW100/RS232 is connected to the supply voltage and the microcontroller is initialized. With a watchdog reset, the Power LED is lit red.

6.1.2 CAN LED

The CAN LED lit green for each message received or transmitted without errors. When the CAN error warning level is reached, the corresponding LED lit red with each reception and transmission. In the 'CAN BUS OFF' state, no more communication is possible and the LED permanently lit red. If data are lost on the CAN interface, the CAN LED flashes red in stand-by mode, with each message received or transmitted without errors the CAN LED briefly lit green in between.

6.1.3 Status LED

In COPmode, the Status LED lit as defined in the "CANopen Indicator Specification":

LED state	Meaning
Is lit green	CANopen state is OPERATIONAL
Flashes green	CANopen state is PREOPERATIONAL
Flickers green	CANopen state is STOPPED
Is lit red	CAN controller is in bus off state
Flashes green-red	Error or warning on CAN controller

The Status LED has no function in the operation mode CANmode.

6.1.4 RS232 LED

The RS232 LED flickers green during serial communication. If data are lost on the RS232 interface, the RS232 LED flashes red in stand-by mode, with serial communication the RS232 LED flickers red.

6.2 Configuration mode

In the configuration mode, the Status LED and the CAN LED flash green simultaneously. The RS232 LED flickers green during transmission of the configuration data.

6.3 Error state

With incorrect parameter values in the flash, all LEDs except the Power LED flash red. In this case the configuration values must be checked and loaded into the device again.

7 Notes on EMC

7.1 Shield concept

The highest interference immunity is achieved when the shield of the CAN bus is grounded on the assembly plate and the ground terminal (pin 1 / PE) of the CAN-GW100/RS232 is connected to the next available grounding (see Fig. 7-1). Via parallel connection of a resistor (1 MΩ) and a capacitor, the grounding connection is connected internally to the GND of the CAN and of the supply voltage.

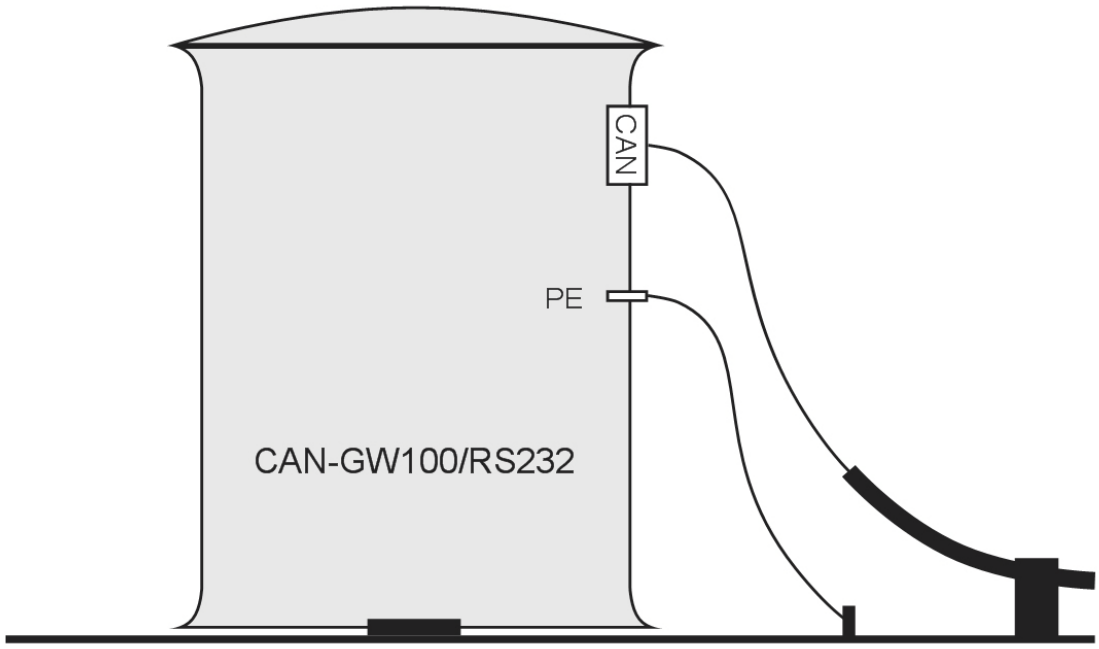


Fig. 7-1: Shield concept CAN-GW/RS232

8 Appendix

8.1 Support

For more information on our products, FAQ lists and installation tips, please refer to the support section of our website (http://www.ixxat.de), which also contains information on current product versions and available updates.

If you have any further questions after studying the information on our website and the manuals, please contact our support department. The support section on our website contains the relevant forms for your support request. In order to facilitate our support work and enable a fast response, please provide precise information on the individual points and describe your question or problem in detail.

If you would prefer to contact our support department by phone, please also send a support request via our website first, so that our support department has the relevant information available.

8.2 Returning hardware

If it is necessary to return hardware to us, please download the relevant RMA form from our website and follow the instructions on this form.

In the case of repairs, please also describe the problem or fault in detail on the RMA form. This will enable us to carry out the repair quickly.

8.3 Technical specifications

Power supply:	9 V – 36 V DC
Power consumption:	approx. 1.5 W
Dimensions:	approx. 120 X 90 X 30 mm
Weight:	approx. 100 g
Temperature range:	Operation: 0 – 50 °C, storage: -40 - 125 °C
Relative humidity:	10 - 95 %, non-condensing
Protection type:	IP20
Galvanic isolation:	as an option CAN-bus to internal logics 250 V AC for 1 min
Delay galv. isolation:	typ. 50 ns
CAN message buffer	100 messages per CAN segment
RS232 message buffer	100 bytes
PDO performance	In the COPmode operation mode, the performance of the device is restricted to 500 PDO per second. This corresponds to a data flow of 19200 bit/s with full duplex or 38400 bit/s with half duplex
EMC test in accordance with:	DIN EN 55022 / 05.1999 DIN EN 61000-4-2 / 03.1996 DIN EN 61000-4-3 / 06.1999 DIN EN 61000-4-4 / 03.1996 DIN EN 61000-4-5 / 09.1996 DIN EN 61000-4-6 / 04.1997
Cable supplied:	Laplink cable (for connection to a PC)

8.4 Sources of data sheets

CAN-Controller SJA1000 and Transceiver 82C251

http://www.philips-semiconductors.com

8.5 EC conformity declaration

IXXAT Automation hereby declares that the product: CAN-GW100/RS232

Model:	CAN-GW100/RS232
with article numbers:	1.01.0033.11000 1.01.0033.22000
fulfils the requirements of the standards:	DIN EN 55022/ 05.1999 class B DIN EN 61000-6-2 / 03.2000
according to the following test report:	IX227_01.DOC IX216_01.DOC

The product thus complies with the EC directive: 89/336/EEC

This declaration applies to all devices that bear the CE symbol and loses its validity if modifications are carried out on the product.

11.09.03, Dipl.-Ing. Christian Schlegel , Managing Director

IXXAT Automation GmbH
Leibnizstr. 15
88250 Weingarten