

# CANblue II

Intelligent Bluetooth/CAN Interface

---



## **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](http://www.ixxat.de)  
E-Mail: [info@ixxat.de](mailto: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](http://www.ixxat.com)  
E-Mail: [sales@ixxat.com](mailto: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](mailto:support@ixxat.de)

### **For customers from the USA/Canada**

Fax: +1-603-471-0880  
E-Mail: [techsupport@ixxat.com](mailto:techsupport@ixxat.com)

## **Copyright**

Duplication (copying, printing, microfilm or other forms) and the electronic distribution of this document is only allowed with explicit permission of IXXAT Automation GmbH. IXXAT Automation GmbH reserves the right to change technical data without prior announcement. The general business conditions and the regulations of the license agreement do apply. All rights are reserved.

## **Registered trademarks**

All trademarks mentioned in this document and where applicable third party registered are absolutely subject to the conditions of each valid label right and the rights of particular registered proprietor. The absence of identification of a trademark does not automatically mean that it is not protected by trademark law.

<b>1</b>	<b>General functionality.....</b>	<b>6</b>
<b>2</b>	<b>Hardware.....</b>	<b>7</b>
	<b>2.1 Features.....</b>	<b>7</b>
	<b>2.2 Connections and control elements.....</b>	<b>7</b>
	2.2.1 Power supply X1 (PWR) .....	7
	2.2.2 CAN bus plug X2 (CAN).....	7
	2.2.3 LED display .....	8
	2.2.4 Pushbutton .....	9
	2.2.5 Bluetooth .....	9
<b>3</b>	<b>Extended ASCII protocol .....</b>	<b>10</b>
<b>4</b>	<b>Behavior of the CANblue II .....</b>	<b>11</b>
	<b>4.1 Restore factory settings .....</b>	<b>11</b>
	<b>4.2 Firmware Update .....</b>	<b>11</b>
	<b>4.3 VCI Support .....</b>	<b>11</b>
	<b>4.4 Bluetooth transmission behavior.....</b>	<b>12</b>
	<b>4.5 Switching the message format.....</b>	<b>12</b>
	<b>4.6 Autostart and handshake .....</b>	<b>12</b>
	<b>4.7 Automatic stop of the CAN controller .....</b>	<b>13</b>
	<b>4.8 CAN filter .....</b>	<b>13</b>
	<b>4.9 Loss of connection .....</b>	<b>14</b>
	<b>4.10 Loss of messages .....</b>	<b>14</b>
	4.10.1 CAN receive buffer overflow.....	14
	4.10.2 Bluetooth transmission buffer overflow.....	14
	4.10.3 CAN transmission buffer overflow .....	14
	4.10.4 Loss of responses to commands.....	14
	<b>4.11 CAN-Controller Errors .....</b>	<b>15</b>
	4.11.1 WARNING: .....	15
	4.11.2 BUS-OFF-Recovery: .....	15
<b>5</b>	<b>Establishing a connection and configuration .....</b>	<b>16</b>
	<b>5.1 Installing the virtual COM port .....</b>	<b>16</b>
	5.1.1 Windows XP .....	16
	5.1.2 Windows 7 .....	20
	<b>5.2 CanBlueCon Configuration Tool .....</b>	<b>24</b>
	5.2.1 Command Line Parameters .....	24

5.2.2	Additional Commands .....	24
5.2.3	Interactive Mode .....	25
5.2.4	Batch Mode .....	26
5.2.4.1	Rx-Tx Demo .....	26
5.2.4.2	Bridge Mode Setup Demo .....	27
<b>5.3</b>	<b>Connecting with Hyperterminal.....</b>	<b>29</b>
<b>5.4</b>	<b>Configuration examples.....</b>	<b>29</b>
5.4.1	Connecting CAN to the PC through the CANblue II .....	29
5.4.2	Configuring a CAN bridge with two CANblue II devices .....	31
5.4.3	Connecting another CANblue II .....	34
<b>6</b>	<b>Extended ASCII-Protocol Commands .....</b>	<b>35</b>
<b>6.1</b>	<b>Device specific Commands .....</b>	<b>35</b>
6.1.1	D VERSION .....	35
6.1.2	D PROTOCOL .....	35
6.1.3	D IDENTIFY .....	36
6.1.4	D INFO .....	36
6.1.5	D CONFIG.....	37
6.1.6	D MAC_ADD .....	39
6.1.7	D MAC_REMOVE .....	39
6.1.8	D MAC_CLEAR .....	40
6.1.9	D MAC_SCAN.....	40
6.1.10	D BUFF_TIMEOUT .....	41
6.1.11	D LINK_POLICY .....	41
6.1.12	D RESET .....	42
6.1.13	D SETTINGS_DEFAULT .....	43
6.1.14	D DISCONNECT_SET .....	43
6.1.15	D DISCONNECT_RESET .....	43
<b>6.2</b>	<b>CAN Controller Commands .....</b>	<b>44</b>
6.2.1	C CONFIG.....	44
6.2.2	C CAN_INFO .....	45
6.2.3	C CAN_INIT .....	46
6.2.4	C CAN_INIT_AUTO .....	47
6.2.5	C CAN_INIT_CUSTOM .....	47
6.2.6	C CAN_START .....	48
6.2.7	C CAN_STOP .....	49
6.2.8	C CAN_RESET .....	49

6.2.9	C AUTOSTART .....	50
6.2.10	C SEND_CAN_FRAMES .....	50
6.2.11	C FILTER_ADD .....	51
6.2.12	C FILTER_REMOVE .....	52
6.2.13	C FILTER_CLEAR.....	52
6.2.14	C FILTER_ENABLE .....	53
6.2.15	C FILTER_DISABLE .....	53
<b>6.3</b>	<b>CAN Bluetooth Messages.....</b>	<b>53</b>
6.3.1	M (ASCII) .....	54
6.3.2	X (Binary) .....	54
<b>6.4</b>	<b>Error Response .....</b>	<b>57</b>
<b>7</b>	<b>Appendix.....</b>	<b>58</b>
7.1	Support .....	58
7.2	Returning hardware .....	58
7.3	Disposing of old equipment .....	58
7.4	Information on EMC .....	58
7.5	Compliance with RoHS directive.....	58
7.6	FCC Compliance .....	59
7.7	Japan Radio Equipment Compliance (TELEC).....	59
7.8	EC Declaration of Conformity .....	61
7.9	Technical Specifications.....	62

# 1 General functionality

The CANblue II units enables multiple CAN networks to connect wirelessly using the Bluetooth Serial Port Profile (SPP). If two CANblue II units are connected, one unit acts as the SPP server and one as the SPP client. The units can act as server and client in different connections simultaneously, allowing more than two units – and thus multiple CAN networks – to be connected. To establish an SPP connection between two units, the unit acting as the client should be given the Bluetooth MAC address of the server. The client then attempts to establish a connection to the other CANblue II. Each CANblue II forwards the messages it receives from the CAN network to all existing SPP connections. Conversely, all CAN messages received from Bluetooth are sent into the CAN network and, if there are any, to other SPP connections as well.

In addition to the SPP server and client for chaining CANblue II units, each CANblue II also provides an additional SPP server. Any other Bluetooth-capable unit that supports SPP can connect to this server. This connection can be used to configure the CANblue II and CAN messages can be received or sent.

There is an ASCII protocol defined for communication with the CANblue II that provides commands for the configuration of the units and the transmission/receiving of CAN messages.

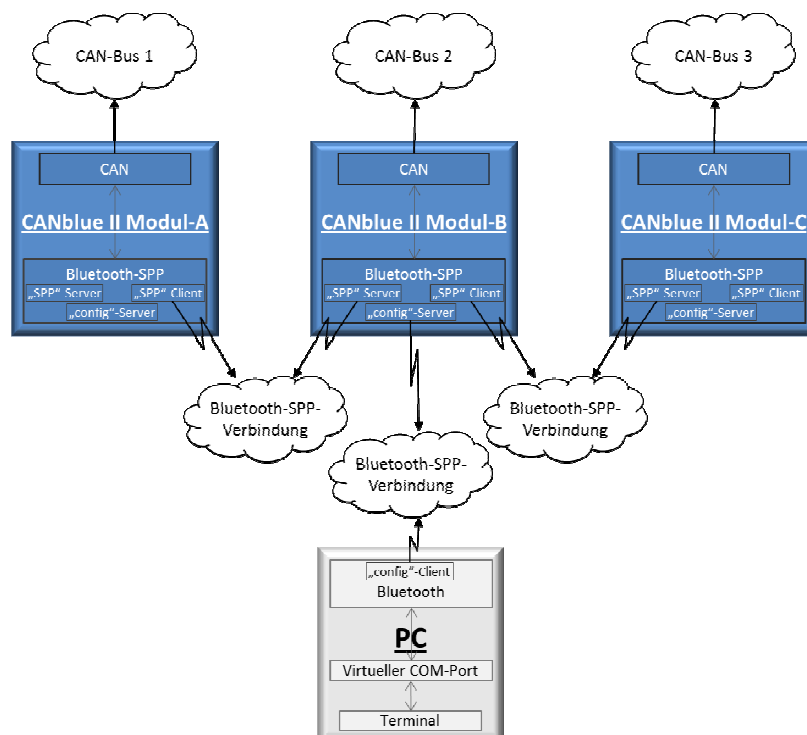


Fig.: 1-1 Networking example

## 2 Hardware

### 2.1 Features

- Bluetooth specification V 2.1 + EDR (Enhanced Data Rate)
- Power supply 9 - 30 V DC
- Microcontroller STM32F103RC with integrated CAN controller, 72 MHz
- Bluetooth Radio Ericsson STLC2500

### 2.2 Connections and control elements

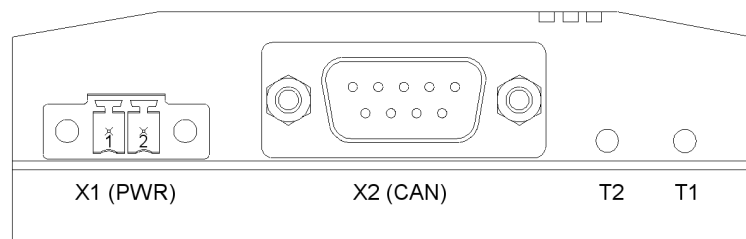


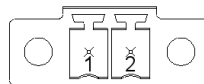
Fig.: 2-1 Connections and control elements

#### 2.2.1 Power supply X1 (PWR)

The unit is supplied with a DC voltage from 9 V to 30 V. The connection pinout is shown in the following table.

The CANblue II is protected against polarity reversal.

X1 Pin no.	Signal
1	PWR (+)
2	GND (-)



#### 2.2.2 CAN bus plug X2 (CAN)

The CANblue II has an ISO 11898-2 bus coupling. The signals for the bus coupling are on the 9-pin sub-D plug as shown in the following table.

X2 Pin no.	Signal
1	-
2	CAN-L
3	GND
4	-
5	-
6	-
7	CAN-H
8	-
9	-



### 2.2.3 LED display

The CANblue II has three LEDs for signaling different states:

LED	Display	Description
Mode	steady red	No Bluetooth MAC address is stored in the configuration of the CANblue II, and there is no connection to an SPP server on the unit.
CAN	flashing green	A CAN message has been sent or received and the CAN controller is not in the warning state.
	flashing red	A CAN message has been sent or received and the CAN controller is in the warning state.
	steady red	The CAN controller is in the BUS-OFF state.
Bluetooth	flashing blue (2 Hz)	An attempt is made to establish a Bluetooth SPP connection with another Bluetooth device or a connection is being established to this device.
	flashing blue (10 Hz)	Bluetooth SPP is used to send or receive data.
	steady blue	There is at least one Bluetooth SPP connection to another device.



### **2.2.4 Pushbutton**

<b>Pushbutton</b>	<b>Description</b>
T1	Restore factory settings see Section 4.1
T2	Not used

### **2.2.5 Bluetooth**

The internal Bluetooth interface needs a unique MAC address (MAC-ID) to communicate. The MAC-ID is on the back of the device and is also used for the unique identification of devices when searching for them with Bluetooth. See also chapter 5, "Establishing a connection and configuration".

### 3 Extended ASCII protocol

To configure and transmit Bluetooth CAN messages, there is an ASCII protocol defined. There is also a binary format available for the transmission of Bluetooth CAN messages to permit a better data rate. CANblue II units always use the binary format for CAN messages transmitted between them.

**ASCII commands have the following structure:**

Message type	Command	Parameter 1	...	Parameter n	LF or CR-LF
--------------	---------	-------------	-----	-------------	-------------

- Individual fields are separated by blanks.
- Multiple sequential blanks are considered to be a single blank.
- There is no distinction between capital and lower-case letters.
- A message is terminated with the ASCII linefeed control code (LF or "\n") or with a carriage return and linefeed (CR LF or "\r\n").
- ASCII messages sent by the CANblue II are terminated with the same ASCII control codes as ASCII messages sent by the user. If the user has not yet sent any ASCII messages, the CANblue II uses CR-LF as the terminator.

There are six different message types defined. The message type is defined by the first byte.

"D"	Device-specific commands
"C"	CAN-specific commands
"M"	CAN messages in ASCII format
"X"	CAN messages in binary format
"I"	Info messages
"E"	Error messages

**Examples:**

ASCII command	Response from the CANblue II
„C CAN_INIT 250\n“	„I OK: CAN_INIT\n“
„C CAN_START\r\n“	„I OK: CAN_START\r\n“
„C FILTER_ADD EXT 7FA1 RTR\r\n“	„I OK: FILTER_ADD\r\n“
„D SETTINGS_DEFAULT\n“	„I OK: SETTINGS_DEFAULT\n“

A list of all ASCII commands can be found in chapter 6 *"Extended ASCII-Protocol Commands"* (p.35).

## 4 Behavior of the CANblue II

### 4.1 Restore factory settings

If there is a "Config" connection to the CANblue II, the device can be reset to the factory settings using the command "D SETTINGS\_DEFAULT".

Without a Bluetooth connection, the device can also be reset as follows:

- (1) Turn off the CANblue II.
- (2) Press and hold button T1
- (3) Turn on the CANblue II; the CAN LED lights in red-green
- (4) When the CAN LED flashes in red-green, release button T1
- (5) If the MODE LED flashes several times, this indicates that the configuration has been reset to factory settings

### 4.2 Firmware Update

Starting with firmware version 2.00.00, an update of the CANblue II firmware is possible. The files needed for updating the CANblue are supplied on CD or can be found in the installation folder. The firmware can be updated as follows:

- (1) Restore the CANblue II to factory defaults
- (2) Set up a virtual COM-Port (Config connection). (see 5.1 Installing the virtual COM port on p. 16)
- (3) Open the command prompt and navigate to the folder "FW-Update" on CD or in the installation folder.
- (4) Call FW-Update <COM-PORT> e.g. "FW-Update 5"
- (5) At first the new Firmware version and the firmware version of the device are displayed, after that the firmware update is performed
- (6) At the end the CANblue II restarts itself. During the restart the CAN and Mode LEDs should flicker red/green
- (7) The firmware version can be checked with CANblueCon and the command „D VERSION“

### 4.3 VCI Support

The CANblue II can also be used as VCI interface starting with firmware version 2.00.05 and VCI Version 3.5.1.3753. Therefore no special firmware has to be flashed. The VCI mode works best when the CANblue II is restored to factory defaults before and it is not running in bridge mode. With reduced receive- and transmit- performance the parallel usage is also possible. Existing CAN filters will be cleared during VCI interface usage and will be restored afterwards.

### 4.4 Bluetooth transmission behavior

With the standard configuration, pending messages are sent immediately by the CANblue II via Bluetooth.

The behavior of the CANblue II can also be changed so that it attempts only to send complete Bluetooth SPP packets. To do this, the command "D BUFF\_TIMEOUT" can be used to specify a time after which pending messages will be transmitted even if they don't complete a full Bluetooth SPP packet. A timeout of 0 indicates that data should be send immediately.

The size of a packet depends on the other node in the connection. CANblue II units used data packets of 669 bytes between themselves.

### 4.5 Switching the message format

The command "C SEND\_CAN\_FRAMES" can be used on a "Config" connection to switch between ASCII and binary format, or the receipt of CAN messages can be disabled entirely (see p. 50, chapter 6.2.10 C SEND\_CAN\_FRAMES).

The format is also changed in the following situations:

- After a connection is established to the "Config" server, the transmission of CAN messages is disabled.
- If the command "C CAN\_START" is issued, the transmission format is switched to ASCII.
- If the "Config" connection is used to sent a CAN message to the CANblue II in the ASCII or binary format, the CANblue II switches to the same format.

If the CANblue II is in autostart mode and a handshake is carried out on the "Config" connection (see 4.6 *Autostart and handshake*), the device switches to the binary format.

### 4.6 Autostart and handshake

If the autostart mode of the CANblue II is enabled (see 6.2.9 C AUTOSTART on p. 50 ) and a SPP connection is established, it attempts to carry out a handshake to start the CAN controller.

If a handshake is carried out between two CANblue II devices, both devices must have autostart mode enabled.

However, a handshake can also be carried out on the "Config" connection. The corresponding responses to the handshake messages must then be sent manually by the user.

**A handshake works as follows:**

After an SPP connection is established, the SPP server transmits its version information (e.g. "I CANblue Generic - Bridge v2.00.03"). The SPP client must then also send its own version information.

If the SPP server receives no response to its version information, it sends the version information again after five seconds.

Once the version information has been exchanged successfully, the SPP server starts its CAN controller at the configured baud rate and sends "I CAN STARTED" to the SPP client. The client then starts its own CAN controller and sends "I CAN STARTED" back to the SPP server.

This concludes the handshake procedure and both CANblue II units will now exchange CAN messages in binary format.

**4.7 Automatic stop of the CAN controller**

If there is no more SPP connection to the CANblue II, the CAN controller automatically stops.

**4.8 CAN filter**

Messages received by the CAN controller can be filtered. Messages are filtered based on the identifier, the frame format (extended, standard), and the frame type (data, remote). Filter entries can be stored in the CANblue II for filtration. An entry consists of the frame format, the ID, and the frame type. Once filtering is activated, messages received by the CAN controller are only forwarded on the SPP connections if the messages correspond to a filter entry.

4096 standard filters can be entered. This includes all possible identifiers for the standard frame format.

For the extended filter, there are 300 bytes of storage available. An extended filter entry occupies 8, 16, 24, or 32 bits depending on the number of CAN ID digits. Thus between 75 and 300 extended messages can be filtered.

<b>CAN-ID range</b>	<b>Memory consumption in bytes</b>
0-7F	1
80-7FFF	2
8000-7FFFFFFF	3
800000-1FFFFFFFFF	4

The following commands are available for configuration of filtering:

„C FILTER\_ADD“ (S.51, Kapitel 6.2.11)

„C FILTER\_REMOVE“ (S.52)

„C FILTER\_CLEAR“ (S.52)

„C FILTER\_ENABLE“ (S.53)

„C FILTER\_DISABLE“ (S.53)

### 4.9 Loss of connection

If a CANblue II has stored a Bluetooth MAC address, then for five seconds it will attempt to establish an SPP connection to that address. If the connection attempt fails, then a new attempt is always started after two seconds.

The loss of an existing SPP connection is detected after three seconds. After connection loss is detected, the SPP client immediately attempts to establish a new connection as described above.

### 4.10 Loss of messages

#### 4.10.1 CAN receive buffer overflow

The CAN receive buffer can overflow if a Bluetooth connection to the CANblue II is established during high traffic on the connected CAN network or another connection is attempted. If this is the case, additional incoming CAN messages are discarded. If there is a config connection to the CANblue II, this is indicated with an error message ("E 84 Rx SW queue OVERRUN").

#### 4.10.2 Bluetooth transmission buffer overflow

The CANblue II has a separate transmission buffer for every SPP connection. If one of these buffers fills up due to an excess number of CAN messages, any additional incoming messages for this buffer are discarded. If this involves the config connect, then once there is space in the buffer again an error message is send ("E 84 Rx SW queue OVERRUN").

#### 4.10.3 CAN transmission buffer overflow

Due to flow control on the Bluetooth SPP connection, the CAN transmission buffer can normally not overflow. However, to avoid blocking the receipt of data on the SPP connections, in case of error (CAN controller in warning or BUS OFF state) or if there are more than 512 messages in the buffer, the oldest buffer entries are overwritten.

#### 4.10.4 Loss of responses to commands

If there is high data traffic between the SPP connections on the CANblue II devices and a command is sent on the config connection, it can occur that parts of the CANblue II response are discarded. Only entire lines of the re-

sponse are discarded, that is, the response is always terminated with a linefeed or carriage return and linefeed.

### 4.11 CAN-Controller Errors

#### 4.11.1 WARNING:

If the CAN controller is in the warning state due to multiple incorrectly received or transmitted messages, this can only be corrected by resetting the CANblue II or by the receipt or transmission of multiple valid CAN messages. The stopping and restarting of the CAN controller does not reset the warning state (except for BUS-OFF).

#### 4.11.2 BUS-OFF-Recovery:

If the CAN controller goes into BUS-OFF, the BUS-OFF recovery is automatically started. Five seconds after detection of the BUS-OFF state, the CAN controller is stopped for one second, and then restarted. If the CAN controller then detects 128 successive 11-bit sequences on the bus (128 valid messages), all error flags are reset on the CAN controller and the CAN controller is then placed back into the normal operating condition. The BUS-OFF recovery is carried out until the CAN controller is in normal operating mode or is stopped through the config connection (see 6.2.7 *C CAN\_STOP*).

The BUS-OFF recovery can also be carried out manually by using the config connect to stop and restart the CAN controller.

# 5 Establishing a connection and configuration

Each CANblue II provides two virtual SPP servers as a service. The names of the SPP servers are "Config" and "SPP". To configure a CANblue II, a Bluetooth-capable device that supports the serial port profile (SPP) must be used to establish a connection to the Config server. To connect to the Config server of a CANblue II, a virtual COM port must be installed for the SPP connection on the device used. The user can see the virtual COM port as a physical COM connection present on the device which is connected through a cable to a CANblue II. The following values must be used for the properties of the COM port.

Baudrate	921600
Data bits	8
Parity Bit	none
Stop bits	1
Flow control	hardware

The COM port can be used with a terminal program, for example. To configure the CANblue II and CAN message exchange, the "Extended ASCII protocol" must be used (see section 3 on p.10 and section 6 *Extended ASCII-Protocol Commands* on p.35).

## 5.1 Installing the virtual COM port

The following two sections describe step by step how a Bluetooth device is added under Windows XP and Windows7 and then used to establish a connection to a CANblue II on a virtual COM port

### 5.1.1 Windows XP

- (1) Open the dialog "Bluetooth devices" (Control Panel → Bluetooth devices). Use the "Add" button to open the Bluetooth device addition wizard.



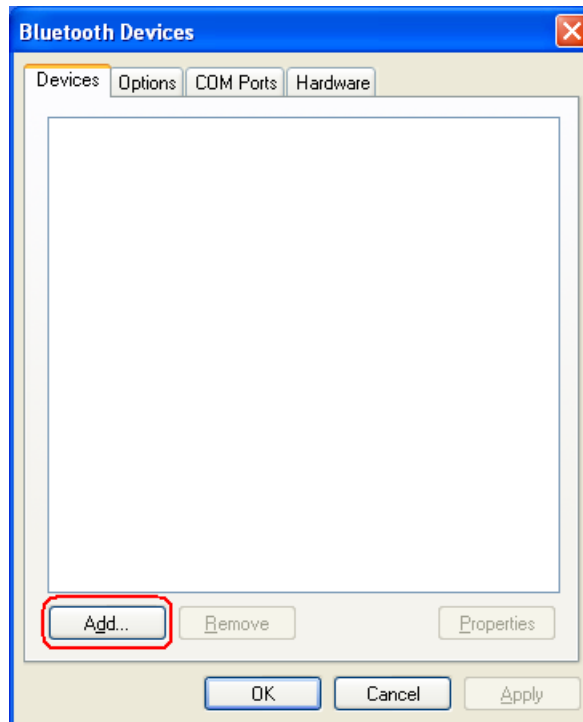


Fig.: 5-1 XP - Bluetooth devices

- (2) Check "My device is set up and ready to be found ", then use the "Next" button to search for devices.



Fig.: 5-2 XP Bluetooth device wizard - Welcome

- (3) All available devices will then be displayed. The CANblue II devices have names like "CANblue II ([MAC address])". The MAC address can be found on the back of the CANblue II. Select the device to which you want to connect and confirm the selection with the "Next" button.

## Establishing a connection and configuration

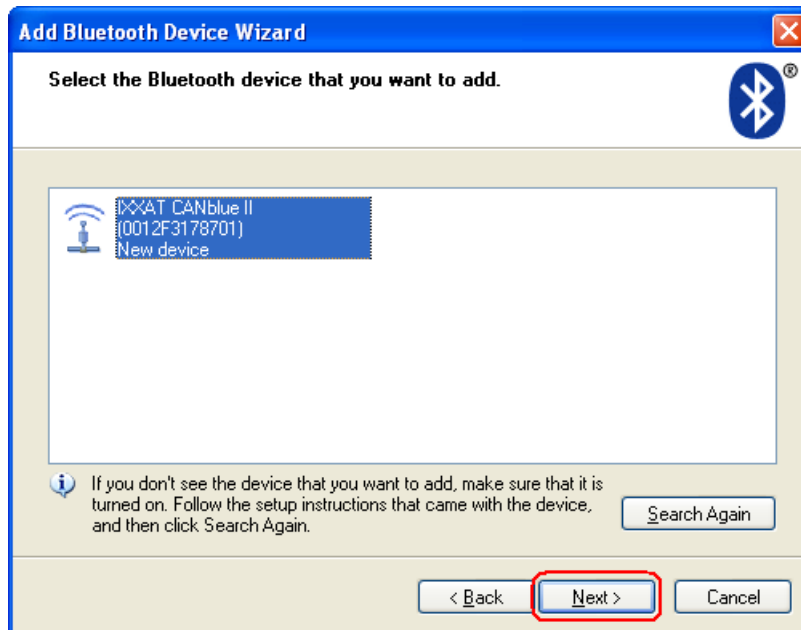


Fig.: 5-3 XP Bluetooth device wizard - Devices found

- (4) Now the passkey for the CANblue II must be entered. "Use the passkey found in the documentation" must be selected for the entry, and "7388" entered as the passkey. Confirm the input with "Next".

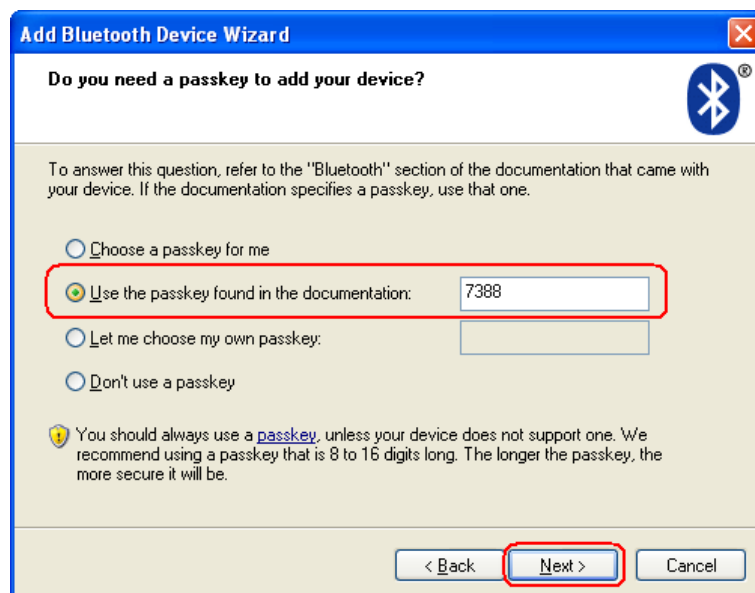
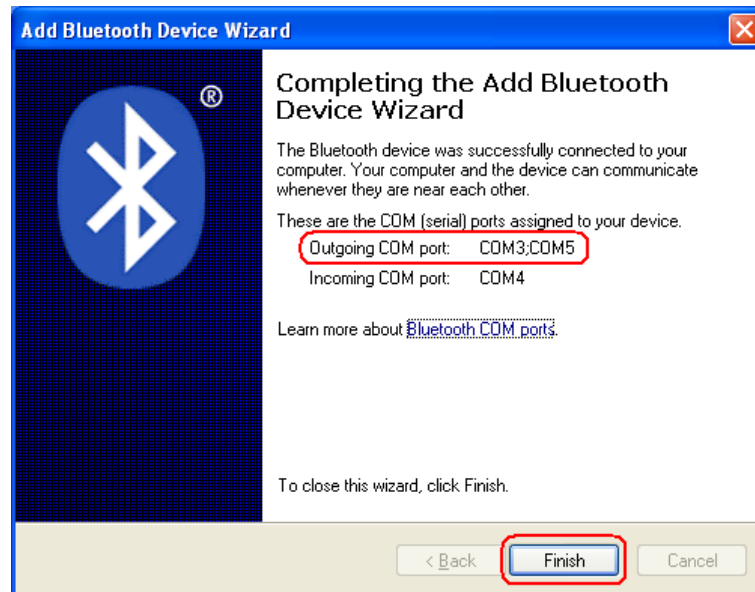


Fig.: 5-4 XP Bluetooth device wizard - passkey

- (5) After all drivers have been installed, the virtual COM ports created for the device are displayed. For the CANblue II devices, two outgoing COM ports are shown. One of these two COM ports is provided for the Config connection of "non-CANblue devices".



**Fig.: 5-5 XP Bluetooth device wizard - Completion**

- (6) To find out which COM port should be used, you must query the names of the SPP servers. In the "Bluetooth Devices" dialog, you will see the CANblue II you just added. Use the "Properties" button to open the "Properties" window for the selected device. Click on the "Services" tab to search for the services of the device and display them. For the CANblue II, the two SPP servers of the device are shown here. One of these servers is named "Config". Next to the name, the COM port is displayed that can be used to establish a connection to the CANblue II. The second service, named "SPP", is reserved for a connection between two CANblue II devices. No connection can be established to this server. If the checkmark is not set for the Config service, there may have been problems installing the driver for this service. Check the box and confirm with the "Apply" button to attempt to install the driver again. An Internet connection may be necessary so that the driver can be downloaded.

## Establishing a connection and configuration

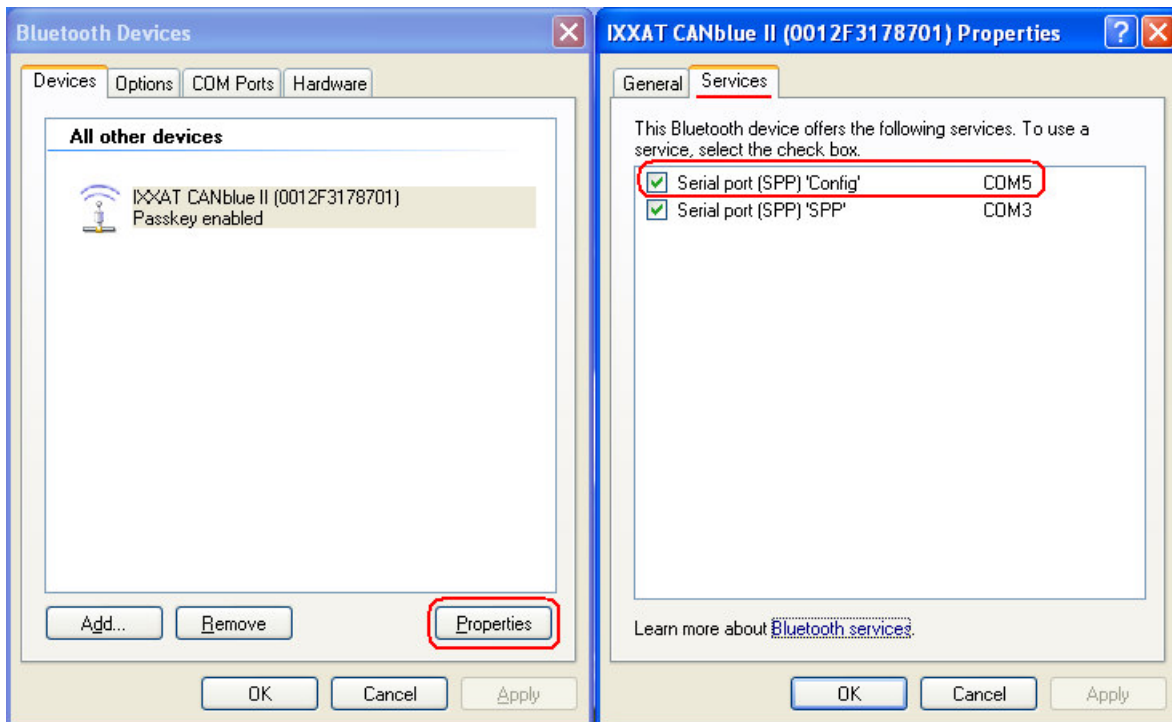


Fig.: 5-6 XP Bluetooth device services

- (7) Now the virtual COM port displayed for the Config connection can be used to connect to the CANblue II.

### 5.1.2 Windows 7

- (1) On the window for "Devices and printers" (Control Panel → "Hardware and Sound" → "Devices and Printer"), the "Add a device" button can be used to search for devices.

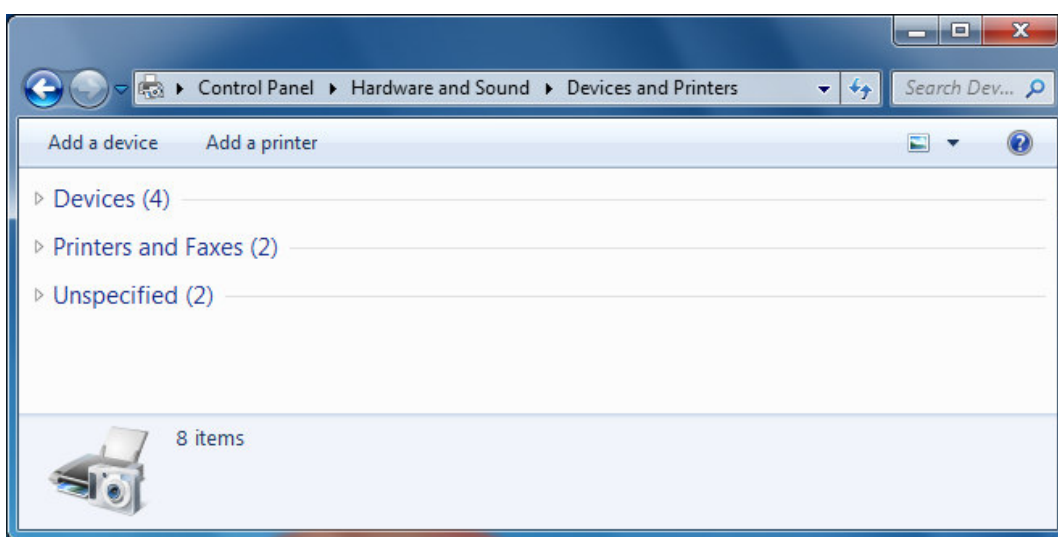
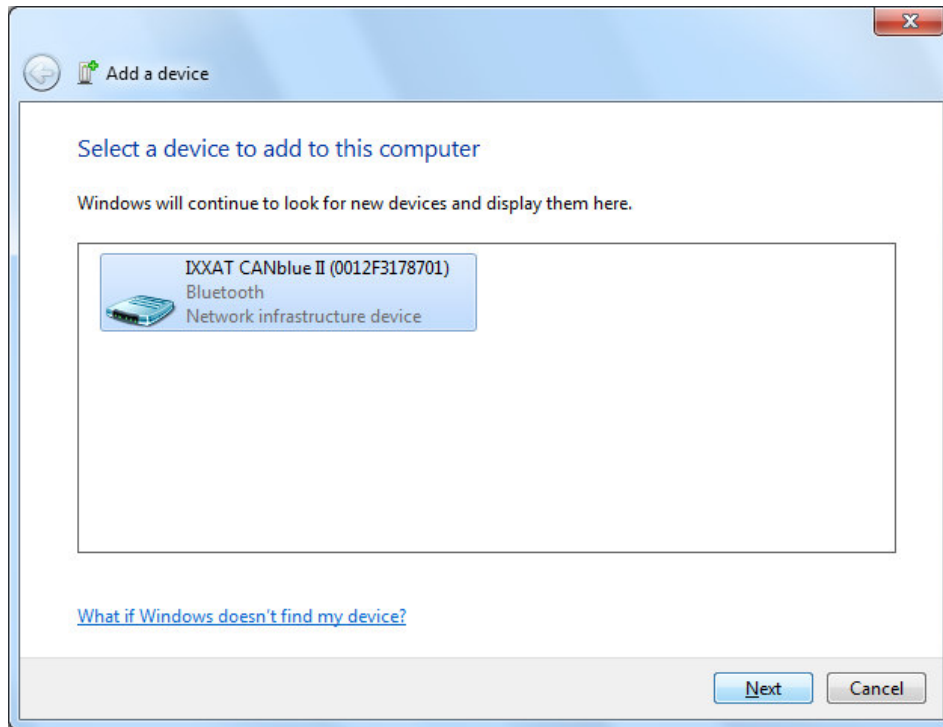


Fig.: 5-7 W7 - Devices and printer

- (2) Select the desired device and confirm with "Next". The CANblue II devices have names like "CANblue II ([MAC address])". The MAC address can be found on the back of the CANblue II. Once the addition is complete, the window can be closed with "Close".



**Fig.: 5-8 W7 - Adding a device**

- (3) Now the pairing code for the CANblue II must be entered. "Enter the device's pairing code" must be selected for the entry, and "7388" entered as the pairing code. Confirm the input with "Next".

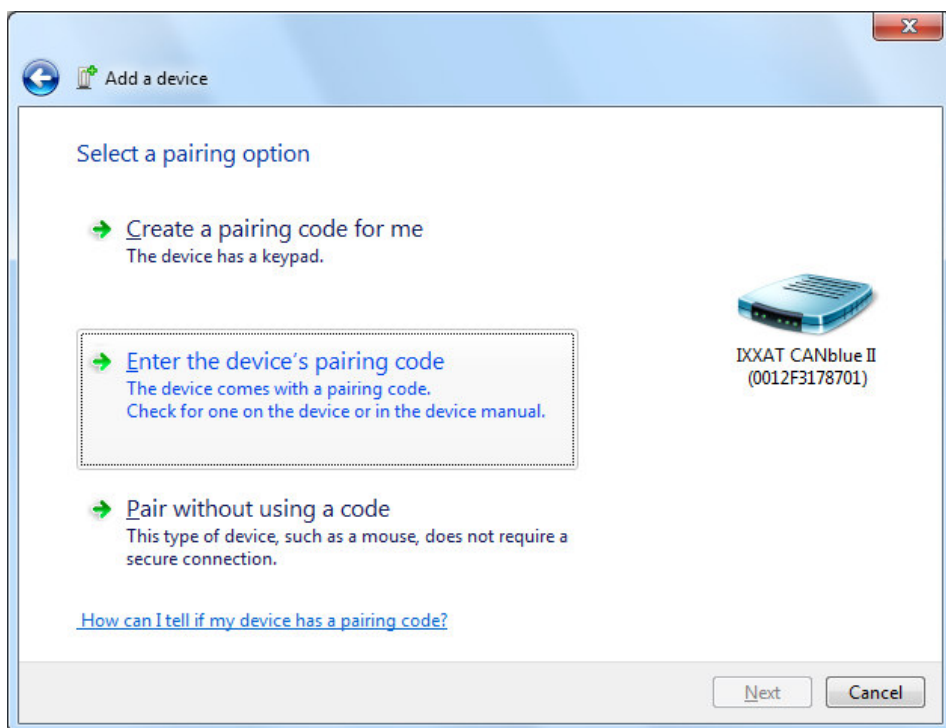


Fig.: 5-9 W7 – Enter device pairing code

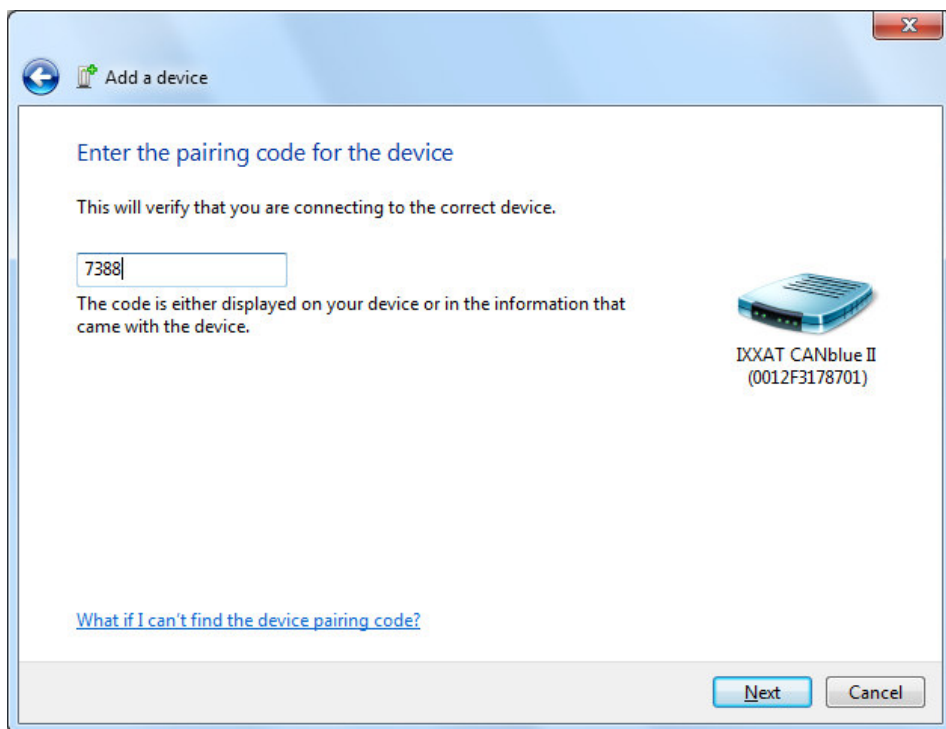


Fig.: 5-10 W7 – pairing code

- (4) The CANblue II added is now displayed on the window for "Devices and Printers". To determine the virtual COM port that can be used to connect to the Config server of the CANblue II, on the Services tab on the properties window for the device (right click on the device → Properties) you can query the services provided by the CANblue II.

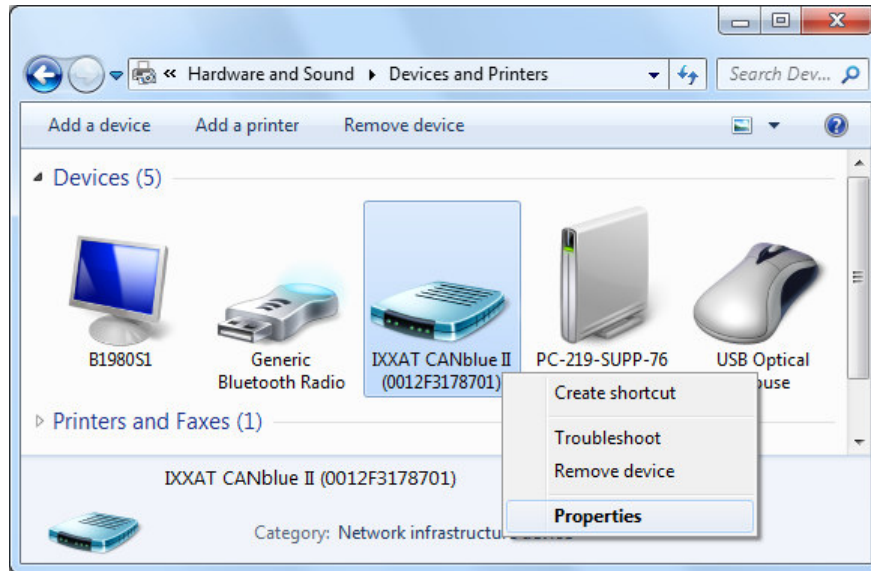


Fig.: 5-11 W7 - Devices and printer - Device Properties

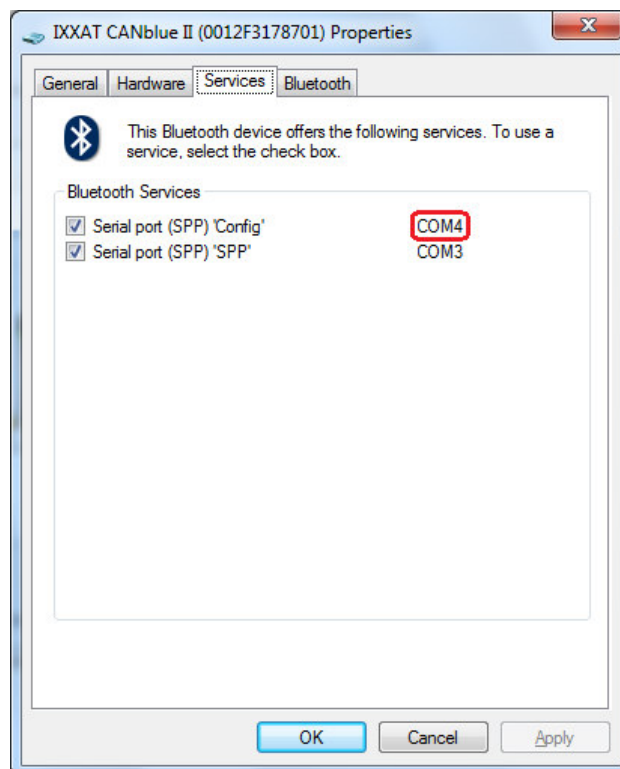


Fig.: 5-12 W7 - Bluetooth device services

- (5) Here is where the two SPP servers and the corresponding COM ports are displayed. The COM port required to connect the PC to the CANblue II is next to the SPP services with the name "Config". If the checkmark is not set for the Config service, there may have been problems installing the driver for this service. Check the box and confirm with the "Apply" button to attempt to install the driver again. An Internet connection may be necessary so that the driver can be downloaded.

## 5.2 CanBlueCon Configuration Tool

To set up the CANblue II, the CanBlueCon tool is provided. It is capable of reading a configuration file with given commands and supports a command history, so that you can scroll through your last issued commands by pressing the UP/DOWN keys.

### 5.2.1 Command Line Parameters

CanBlueCon.exe COM\_PORT\_NUMBER (Console input)

CanBlueCon.exe COM\_PORT\_NUMBER FILENAME (Input from file)

#### Example

**CanBlueCon.exe 4 Config.txt**

The first parameter is mandatory. If you omit the second parameter "**FILENAME**", CanBlueConfig will automatically start up in interactive mode. If you specify a file in the second parameter, CanBlueConfig will start in batch mode.

In interactive mode, CanBlueConfig expects command input from the console, in batch mode it will read the commands from the specified text file.

### 5.2.2 Additional Commands

Additionally to the CANblue commands described in chapter 6 "Extended ASCII-Protocol Commands", the CanBlueCon supports the following "local" commands which start with a "#" character. These commands are interpreted locally and allow the user to implement a cyclic transmission, for instance.



The following additional commands are available:

Command	Parameter	Description
#delay	<DELAY_TIME>	delay in execution for specified time (in sec)
#goto	<LABEL_NAME>	continue execution from string where label is defined
#help	-	print a help screen
#label	<LABEL_NAME>	define label
#pause	-	wait until any key pressed
#print	<TEXT>	print text on display
#exit	-	abort the program

An example of a CANblue II command with local echo and CANblue II reply:

```
>c can_init 1000  
I OK: CAN_INIT
```

An example of a local command with local output:

```
>#print CANblue Generic  
# CANblue Generic
```

### 5.2.3 Interactive Mode

The basic usage of CanBlueConfig in Interactive mode is equal to HyperTerminal. Just type in the command you want to execute and press return. However, CanBlueConfig provides some additional commands to control the execution of the standard commands.

For a reference to these commands see section 5.2.2 Additional Commands.

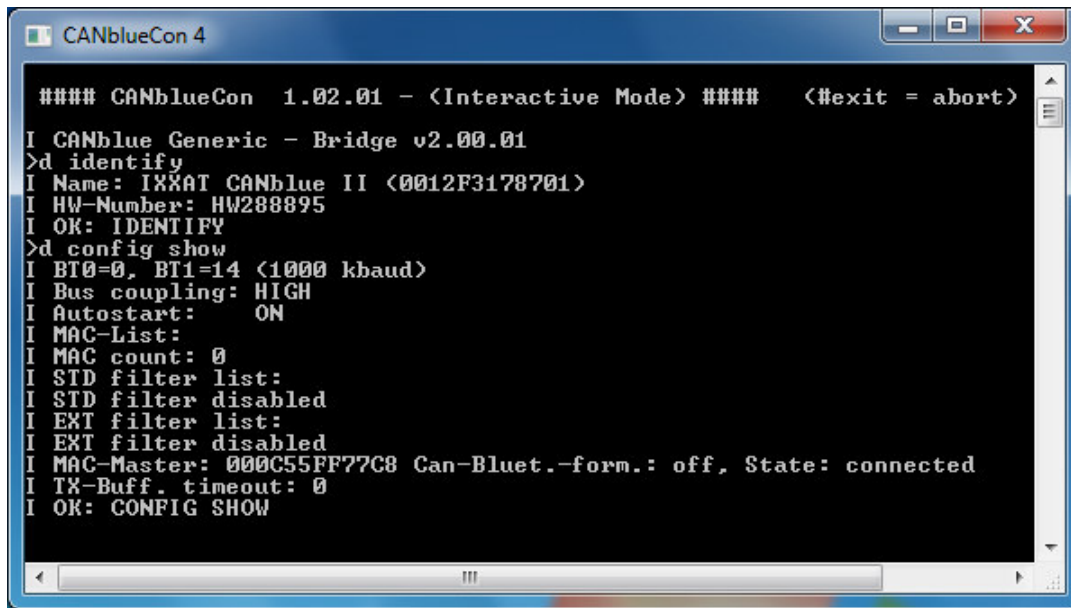


Figure 5.2-1: CANblueCon

### 5.2.4 Batch Mode

You can activate batch mode by specifying a configuration file as the second command line parameter of CanBlueCon.exe. The file is read and the commands are sent to the CANblue II.

The additional commands of CanBlueConfig are especially useful here, because you can easily implement loops and other constructs with them.

#### 5.2.4.1 Rx-Tx Demo

The following example is a part of the configuration file "CanBlueCon\_CAN\_RX\_TX\_Demo.txt".

```
#print Initializing CAN with 1000 kbaud
c can_init 1000

#print
#print Starting CAN controller
c can_start

#print Start loop demo
#pause

#label L1
#print
m sd8 100 1 2 3 4 5 66 77 88
m sd2 101 1 2
m sr0 102
m sr8 103

m ed3 FFF0 FF EE DD
m ed5 FFF1 11 22 33 44 55
m er0 FFF2
```

```
m er8 FFF3
#delay 1
#goto L1
```

If you pass that configuration file to CanBlueCon.exe, by typing “canBlueCon 5 CanBlueCon\_CAN\_RX\_TX\_Demo.txt” (where “5” is the number of the virtual com port, and “CanBlueCon\_CAN\_RX\_TX\_Demo.txt” the name of the configuration file), you can see the following output:

```
C:\CanBlueCon>CanBlueCon 5 CanBlueCon_CAN_RX_TX_Demo.txt
```

```
# Initializing CAN with 1000 kBaud
> c can_init 1000
I OK: CAN_INIT
#
# Starting CAN controller
> c can_start
I OK: CAN_START
# Start loop demo
press any key...
```

After pressing a key, 8 CAN messages are transmitted endlessly with a delay of one second:

```
> m sd8 100 1 2 3 4 5 66 77 88
> m sd2 101 1 2
> m sr0 102
> m sr8 103
> m ed3 FFF0 FF EE DD
> m ed5 FFF1 11 22 33 44 55
> m er0 FFF2
> m er8 FFF3
```

### 5.2.4.2 Bridge Mode Setup Demo

In the following, two sample configuration files for a bridge mode set-up are shown.

“CanBlueCon\_Initialization\_Master.txt”:

```
#print #####
#print ## Demo: CANblue master initialization ##
#print #####

#print
#print Showing CANblue Generic Version
d version
#delay 0.5

#print
#print Resetting device to factory default
d settings_default
```

## Establishing a connection and configuration

---

```
#print
#print Initializing CAN with 1000 kBaud
c can_init 1000

#print
#print Setting AUTOSTART parameter to on, this is important for establish-
ing a CANblue Bridge
c autostart on

#print
#print Adding MAC address for establishing a CANblue Bridge
#print Change the MAC address according to your slave device
d mac_add 112233445566

#print
#print Showing current configuration
c config show
#delay 0.5

#print
#print Saving the configuration
c config save

"CanBlueCon_Initialization_Slave.txt":
#print #####
#print ## Demo: CANblue slave initialization ##
#print #####

#print
#print Showing CANblue Generic Version
d version
#delay 0.5

#print
#print Resetting device to factory default
d settings_default

#print
#print Initializing CAN with 1000 kBaud
c can_init 1000

#print
#print Setting AUTOSTART parameter to on, this is important for establish-
ing a CANblue Bridge
c autostart on

#print
#print Showing current configuration
c config show
#delay 0.5

#print
#print Saving the configuration
c config save
```

You have to pass those configuration files to CanBlueCon.exe to set up both CANblue II devices by typing

"CanBlueCon 5 CanBlueCon\_Initialization\_Master.txt" and

"CanBlueCon 6 CanBlueCon\_Initialization\_Slave.txt".


The numbers "5" and "6" are the numbers of the virtual com ports.



The bridge master must have configured the MAC address (here "112233445566") of the bridge slave, so that the master can establish a Bluetooth connection to the slave. Please, adapt the MAC address of the example to your slave device. The slave on the other hand must not contain a configured MAC address in its MAC address list.

### 5.3 Connecting with Hyperterminal

The following describes how Hyperterminal can be used to connect to the CANblue II using the COM port just installed.

- (1) After starting Hyperterminal, you must assign the connection a name. This name can be any arbitrary name; confirm with "OK"..
- (2) In the next dialog, the COM port must be selected for the Config connection to the CANblue II. The COM port can be determined as described in section 5.1 *Installing the virtual COM port* under point 6. After confirming with "OK", the PC attempts to connect to the CANblue II. If the connection attempt fails, the  button can be used to reestablish the connection.
- (3) Now every character entered on the keyboard is sent to the CANblue II and characters sent by the CANblue II are displayed in the Hyperterminal window. The CANblue II processes incoming messages only when it receives a linefeed or a carriage return followed by a linefeed.
- (4) To send a carriage return and linefeed at the end of an entered command by pressing the Enter key, the box "Terminate transmitted lines with a linefeed" must be checked under "File" → "Properties" → "Settings" tab → "ASCII configuration". For better clarity, the "Output characters entered locally (local echo)" can be checked as well.

### 5.4 Configuration examples

The following three examples describe how CANblue II devices can be configured for different requirements.

#### 5.4.1 Connecting CAN to the PC through the CANblue II

The following example describes how a previously installed virtual COM port can be used to configure the CANblue II to exchange data with a CAN network connected to the CANblue II. Communication through the COM port can, for example, be carried out with Hyperterminal as described in chapter 5.3.

## Establishing a connection and configuration

The following specifications apply in the example:

- The CAN network is operated at a data rate of 500 kBaud.
- Only the following CAN message should be forwarded by the CANblue II device A:
  - Data and remote frames with standard identifier 5
  - Remote frames with standard identifier 1F.
  - Data frames with extended identifier 1A2B3C

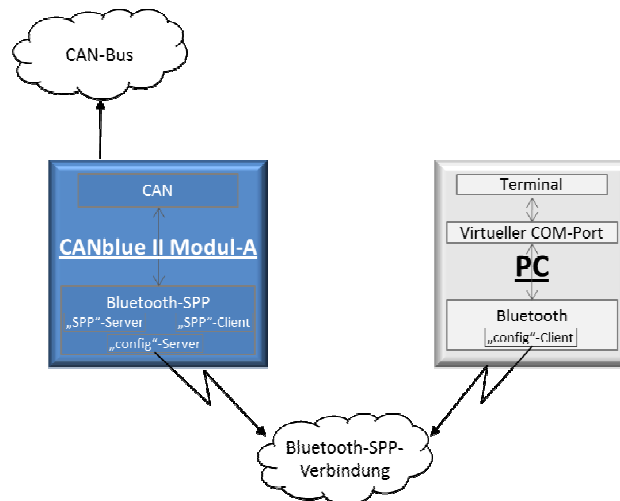


Fig.: 5-13 CANblue II – PC connection

- (1) The command **"D SETTINGS\_DEFAULT↵"** is used to reset the device to factory settings.
- (2) **"C CAN\_INIT 500↵"** initializes the CAN controller to 500 kBaud.
- (3) Setting the filter:
  - **"C FILTER\_ADD 5↵"**
  - **"C FILTER\_ADD STD 5 RTR↵"**
  - **"C FILTER\_ADD STD 1F↵"**
  - **"C FILTER\_ADD EXT 1A2B3C↵"**
- (4) **"C FILTER\_ENABLE STD↵"** activates the standard filter.
- (5) **"C FILTER\_ENABLE EXT↵"** activates the extended filter.
- (6) The **"C CONFIG SHOW↵"** command can be used to check the configuration.
- (7) **"C CONFIG SAVE↵"** saves the current configuration.
- (8) The command **"C CAN\_START↵"** then starts the CAN controller. If the CAN controller receives a message from the CAN network that matches one of the filters entered, it will be transmitted on the Bluetooth SPP connection in ASCII format.
- (9) To send CAN messages to the CANblue II or into the connected CAN network, the ASCII (see 6.3.1 *M (ASCII)*) or binary (see 6.3.2 *X (Binary)*) for-

mat can be used. The CANblue II matches the transmission format of CAN messages on the SPP connection to the format received.

(10) The following command sends a CAN data frame with standard identifier 7FF and the 7 data bytes "1A 2B 3C 4D 5E 6F 70" to the CAN bus:

„M SD7 7FF 1A 2B 3C 4D 5E 6F 70↵“

### 5.4.2 Configuring a CAN bridge with two CANblue II devices

The device A configured as shown in 4.5 should now connect to a second CANblue II device B.

#### Specifications:

- The CANblue II device B is connected to a 1000 kBaud CAN network.
- CANblue II device A should forward all standard CAN messages and filter out all extended CAN messages.
- CANblue II device B should forward all CAN messages.

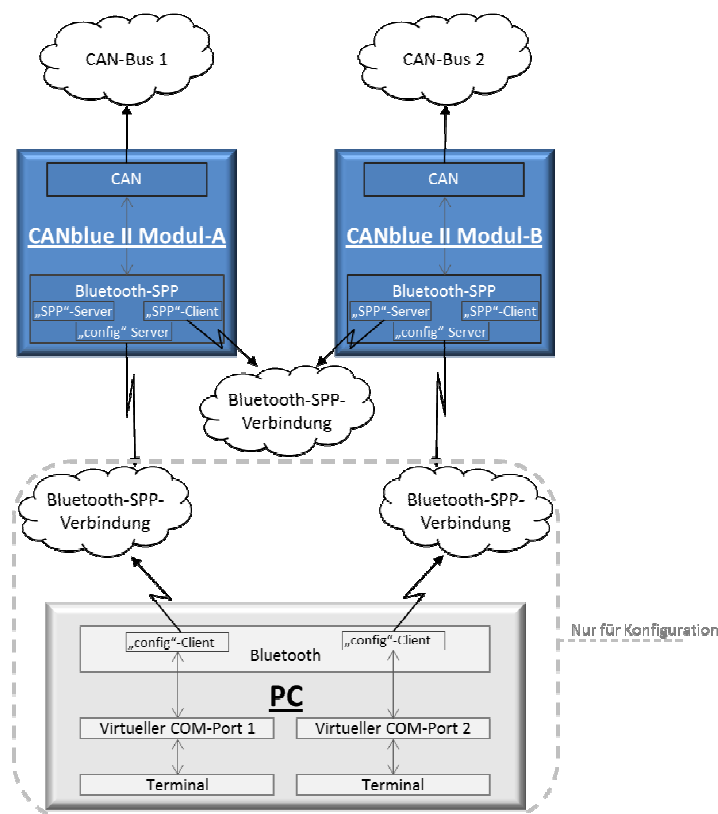


Fig.: 5-14 CANblue II - Bridge

To connect to device B, a virtual COM port as described in 5.1 must be installed and a connection established to it.

## Establishing a connection and configuration

---

- (1) The command **"D SETTINGS\_DEFAULT↵"** is used to reset device B to factory settings. The CAN controller is initialized to 1000 kBaud in the factory settings. **"C CAN\_INIT↵"** need therefore not be issued.
- (2) So that device B can automatically start the CAN controller after connected to another CANblue II, the **"C AUTOSTART ON↵"** command must be used to enable autostart mode.
- (3) **"C CONFIG SAVE↵"** saves the configuration.
- (4) If it does not exist, the connection to the virtual COM port of the Config connection of the CANblue II device A must be reestablished.
- (5) If device A is currently sending CAN messages on the Config connection, we recommend turning it off to simplify configuration. To turn off the transmission of CAN messages, you can either stop the CAN controller with **"C CAN\_STOP↵"** or you can use **"C SEND\_CAN\_FRAMES OFF↵"** to disable the transmission of CAN messages on that specific connection.
- (6) To instruct the CANblue II device A to forward all standard CAN messages, the filtering of standard identifiers must be disabled with **"C FILTER\_DISABLE STD↵"**. To keep the CANblue II from sending extended CAN messages, the **"C FILTER\_CLEAR EXT↵"** command must be used to delete all extended filter entries. The filtering of extended identifiers is still enabled due to the **"C FILTER\_ENABLE EXT"** in the previous configuration.
- (7) To allow the CAN controller to start automatically, **"C AUTOSTART ON↵"** must be used to enable autostart mode.
- (8) The command "



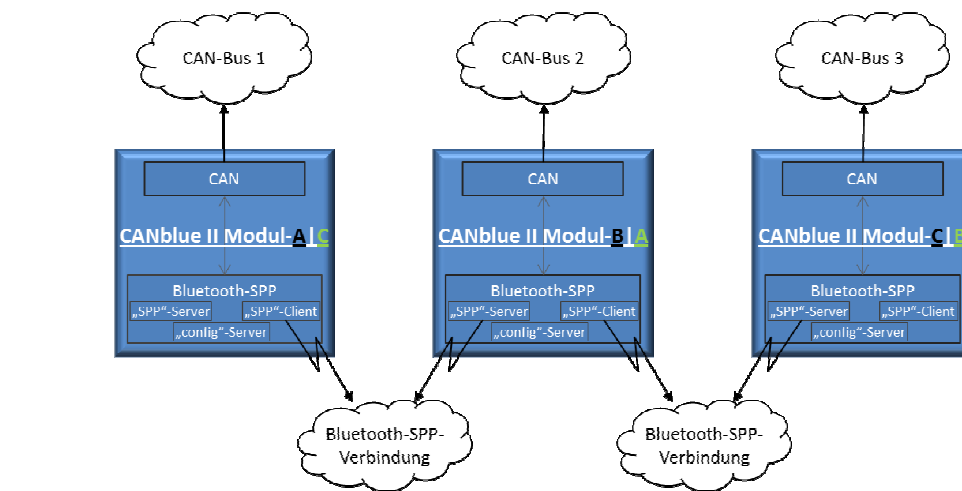
**D MAC\_ADD [address of device B] ↵ "** tells device A to connect to device B. Since autostart mode is enabled on both CANblue II units, the CAN controllers of both CANblue II units are started automatically after connecting. The two CANblue II units will now function as a bridge between the two CAN networks.

- (9) The **"C CONFIG SAVE ↵ "** command can be used to save the configuration.
- (10) To achieve the highest possible data rate between the CANblue II units, the Config connection to the PC should be disconnected. Since the configuration is stored on both units, even if the devices are turned off and on they will reconnect without additional configuration and resume forwarding of their CAN messages.

### 5.4.3 Connecting another CANblue II

To connect a third CAN bus using an additional CANblue II device C with the CAN buses of CANblue II A and B configured in 5.4.2, there are two different options:

- From device B, which has so far only acted as an SPP server, a connection can be established to the SPP server of device C.
- From device C, a connection can be made to the SPP server of device A, which has so far only acted as an SPP client.



**Fig.: 5-15 CANblue II - Bridge chain**

It should be noted that each additional CANblue II increases the rate of CAN messages on the Bluetooth-SPP connections and simultaneously reduces the maximum possible data rate of all SPP connections with each additional SPP connection added.

## 6 Extended ASCII-Protocol Commands

### 6.1 Device specific Commands

#### 6.1.1 D VERSION

D	VERSION	LF/CR-LF
---	---------	----------

Example: „D VERSION↵“

Parameter:

-

Description:

Gets the firmware version of the CANblue II.

Response:

„I CANblue Generic - Bridge v2.00.03\n“

„I OK: VERSION\n“

Errors:

-

#### 6.1.2 D PROTOCOL

D	PROTOCOL	LF/CR-LF
---	----------	----------

Example: „D PROTOCOL↵“

Parameter:

-

Description:

Gets the ASCII protocol version.

Response:

„I ASCII Extended Protocol v1.2\n“

„I OK: PROTOCOL\n“

Errors:

-

### 6.1.3 D IDENTIFY

D	IDENTIFY	LF/CR-LF
---	----------	----------

Example: „D IDENTIFY↵“

Parameter:

-

Description:

Gets hardware version number and name of the CANblue II. The device name contains the Bluetooth MAC address. All LEDs of the CANblue II are blinking.

Response:

„I Name: IXXAT CANblue II (1A2B3C4D5E6F)\n”

„I HW-Number: HW999999\n“

„I OK: IDENTIFY\n“

Errors:

-

### 6.1.4 D INFO

D	INFO	LF/CR-LF
---	------	----------

Example.: „D INFO↵“

Parameter:

-

Description:

Shows information to the actual configured Bluetooth link settings and the current Bluetooth connections. For each connection additional information such as link quality, receive signal strength, or transmission power are shown.

### Response:

Response	Description
„I Link-policy parameter:“	Bluetooth link settings
„I Settingname: DEFAULT“	Name of the configured link settings. See also 6.1.11 D LINK_POLICY
„I Packettype: CC18“	Configured Bluetooth packet types
„I PagescanInterval: 800“	Configured page scan interval
„I PagescanWindow: 12“	Configured pages can window
„I PagescanType: 0“	Configured page scan type
„I Latency (wished): 40“	Wished max. Bluetooth latency in Bluetooth time slots of 625 µs
I Tx-Power (max): 14 dBm“	Maximum allowed Bluetooth transmission power
„I MAC, Latency, Link quality, RSSI, Tx-Power, PacketType“	Table of current Bluetooth connections
„I 123456789ABC, 40*625us, 100%, 15 dB, 1 dBm, CC18“	Table entry of one connection. Information of the connection from left to right: MAC Address, latency in µs, link quality in %, receive signal strength indication in dB (between -127 dB and +128 dB), transmission power in dBm (between -18 dBm and +14 dBm), used Bluetooth packet types
„I OK: INFO“	

### Errors:

-

## 6.1.5 D CONFIG

D	CONFIG	OPERATION	LF/CR-LF
---	--------	-----------	----------

Example: “D CONFIG SHOW↵ ”

### Parameter:

Name	Value	Description
OPERATION	SAVE / LOAD / SHOW	Select one of the values as operation

### Description:

**SAVE:** Saves the current configuration. Saving can take several seconds.  
**LOAD:** Loads a configuration, if existing.  
**SHOW:** Shows the current configuration.

## Extended ASCII-Protocol Commands

---

Response:

**SAVE:**

„I OK: CONFIG SAVE\n“

**LOAD:**

„I OK: CONFIG LOAD\n“

**SHOW:**

Response	Description
„I BT0=0, BT1=14 (1000 kBaud)\n“	Values of the bus timing registers. The name of the configuration is given in brackets.
„I Bus coupling: HIGH\n“	Configured bus coupling. Only HIGH is supported.
„I Autostart: ON\n“	Auto start mode ON/OFF
„I MAC-List\n“	Obsolete. Response MAC-List is used by old CANblue Module.
„I MAC count: 0\n“	Obsolete, see „I MAC-LIST“
„I STD filter list\n“	Content of standard ID filter list.
„I CAN Id: 1\n“	
„I CAN Id: 4, RTR bit set\n“	
„I STD filter enabled\n“	
„I EXT filter list: \n“	Content of extended ID filter list.
„I CAN Id: 4, RTR bit set\n“	
„I CAN Id: 7FFFF\n“	
„I EXT filter disabled\n“	
„I MAC-Slave: 001122334455 Can-Bluet.-form.: binary, State: disconnected\n“	Information on a SPP server connection: MAC-Address, format of CAN messages (ASCII ,BINARY, OFF), connection status (connected, dis- connected).
„I MAC-Master: C44619F9813A Can-Bluet.-form.: off, State: con- nected\n“	Information on a SPP client.
„I TX-Buff. timeout: 0\n“	Timeout value of the send buffer.
„I OK: CONFIG SHOW\n“	

Errors:

Error Response	Description
„E 63 Error while saving config\n“	Error occurred during saving of the configuration. Configuration is lost.
„E 61 No valid config\n“	There is no valid configuration.

### 6.1.6 D MAC\_ADD

D	MAC_ADD	ADR	RECON	LF/CR-LF
---	---------	-----	-------	----------

Example: „D MAC\_ADD 001122334455↵ “

Parameter:

Name	Value	Description
ADR	6 Byte Hexadecimal	Bluetooth MAC address of a second CANblue or CANblue II Module.
RECON	0 – 1000 decimal (Optional, Default: 0)	Parameter is obsolete.

Description:

Adds a Bluetooth MAC address of a SPP server to the CANblue II. The device tries to establish a connection to the SPP server of a Bluetooth device with the added MAC address.

Response:

„I OK: MAC\_ADD\n”

Errors:

Error Response	Description
“E 51 MAC-list is full\n”	Only one MAC address is supported.
„E 53 MAC Address already exists\n”	The MAC address is already used for a connection to a SPP server.

### 6.1.7 D MAC\_REMOVE

D	MAC_REMOVE	ADR	LF/CR-LF
---	------------	-----	----------

Example: „D MAC\_REMOVE 001122334455↵ “

Parameter:

Name	Value	Description
ADR	6 Byte Hexadecimal	MAC address, which should be deleted in the connection list.

Description:

Removes a MAC address from the connection list of the CANblue II. If there is an active connection to another device or the CANblue II currently tries to connect to another device, then this connection is closed before the MAC address is removed. This may lead to a delayed response up to 5 seconds.

## Extended ASCII-Protocol Commands

---

Response:

„I OK: MAC\_REMOVE\n”

Errors:

Error Response	Description
“E 52 Wrong MAC Address\n”	The MAC address is not valid or not within the connection list.

### 6.1.8 D MAC\_CLEAR

D	MAC_CLEAR	LF/CR-LF
---	-----------	----------

Example: „D MAC\_CLEAR↵ “

Parameter:

-

Description:

Deletes all MAC addresses from the connection list of the CANblue II. (In the current firmware, there is only one MAC address supported). If there is an active connection to another device or the CANblue II tries to connect to another device, then this connection is closed before the MAC address is removed. This may lead to a delayed response up to 5 seconds.

Response:

„I OK: MAC\_REMOVE\n”

Errors:

-

### 6.1.9 D MAC\_SCAN

D	MAC_SCAN	TIME	LF/CR-LF
---	----------	------	----------

Example: „D MAC\_SCAN 20↵ “

Parameter:

Name	Value	Description
TIME	1-255 decimal (Optional, Default: 10)	Scan time given in seconds.

Description:

Starts a scan for other Bluetooth devices. After the scan time has expired, the response lists all active devices with their names and Bluetooth MAC addresses. For every device, the response could be delayed up to 5 seconds due to a device name query. A maximum of 10 devices can be displayed.



Response:

```
„I MAC-Address,   Name\n“  
„I 001122334455   Device 1“  
„I 010203040506   IXXAT CANblue(010203040506)“  
„I 012345678901   Mobile Phone X“  
„I 010203040507   IXXAT CANblue II(010203040507)“  
„I OK: MAC_SCAN“
```

Errors:

-

**6.1.10 D BUFF\_TIMEOUT**

D	BUFF_TIMEOUT	TIME	LF/CR-LF
---	--------------	------	----------

Example: „D BUFF\_TIMEOUT 20↵ “

Parameter:

Name	Value	Description
TIME	0-1000 decimal	Time is given in milliseconds. After the timeout, Bluetooth data in the buffer will be send.

Description:

Sets a timeout for the send buffer of the CANblue II (see 4.4 Bluetooth transmission behavior). The timeout is applied for all Bluetooth SPP connections of the device.

The minimum time resolution of the timeout value is 10ms.

Response:

```
„I OK: BUFF_TIMEOUT\n“
```

Errors:

-

**6.1.11 D LINK\_POLICY**

D	LINK_POLICY	CONF	LF/CR-LF
---	-------------	------	----------

Example: „D LINK\_POLICY SHORTEST\_LATENCY↵ “

## Extended ASCII-Protocol Commands

### Parameter:

Name	Value	Description
CONF	<ul style="list-style-type: none"><li>▪ DEFAULT</li><li>▪ SHORTEST_LATENCY</li><li>▪ QUICKEST_CONNECTION</li><li>▪ MOST_ROBUST_CONNECTION</li></ul>	<p>Select one of the predefined Bluetooth configuration, which is applied to all Bluetooth connections.</p> <p><b>DEFAULT:</b> balanced configuration suitable for more than one SPP connection in parallel and for “none” CANblue II devices.</p> <p><b>SHORTEST_LATENCY:</b> the latency for Bluetooth messages is reduced. This setting also reduces the data rate slightly. <u>If this setting is used, there is only one SPP connection possible per CANblue II.</u></p> <p><b>QUICKEST_CONNECTION:</b> allows faster establishment of a Bluetooth bridge. This setting also increases the power consumption of the CANblue II and reduces the data rate.</p> <p><b>MOST_ROBUST_CONNECTION:</b> allows bridging a long distance and the Bluetooth connection is more insusceptible to disturbances. This setting also reduces the data rate slightly.</p>

### Description:

Sets the properties of the Bluetooth connection. To get the best results, both devices of a Bluetooth SPP connection should have the same settings.

### Response:

„I OK: LINK\_POLICY\n“

### Errors:

-

## 6.1.12 D RESET

D	RESET	LF/CR-LF
---	-------	----------

Example: „D RESET↵“

### Parameter:

-

### Description:

The CANblue II sends the response and resets itself. Any established Bluetooth connections are lost.

Response:

„I OK: RESET\n“

Errors:

-

### **6.1.13 D SETTINGS\_DEFAULT**

D	SETTINGS_DEFAULT	LF/CR-LF
---	------------------	----------

Example: „D SETTINGS\_DEFAULT↵ “

Parameter:

-

Description:

The configuration is reset to the factory default values. A stored configuration is deleted.

Response:

„I OK: SETTINGS\_DEFAULT\n“

Errors:

-

### **6.1.14 D DISCONNECT\_SET**

Description:

This command is obsolete. It can be used only with old CANblue devices.

Response:

-

Errors:

Error Response	Description
„E 3 Unsupported command\n“	Command is obsolete.

### **6.1.15 D DISCONNECT\_RESET**

Description:

This command is obsolete. It can be used only with old CANblue devices.

Response:

-

Errors:

Error Response	Description
„E 3 Unsupported command\n“	Command is obsolete.

### 6.2 CAN Controller Commands

#### 6.2.1 C CONFIG

C	CONFIG	OPERATION	LF/CR-LF
---	--------	-----------	----------

Example: "C CONFIG SHOW↵"

Parameter:

Name	Value	Description
OPERATION	SAVE / LOAD / SHOW	Select one of the values as operation

Description:

**SAVE:** Saves the current configuration. Saving can take several seconds.

**LOAD:** Loads a configuration, if existing.

**SHOW:** Shows the configuration.

Response:

**SAVE:**

„I OK: CONFIG SAVE\n“

**LOAD:**

„I OK: CONFIG LOAD\n“

## SHOW:

Response	Description
„I BT0=0, BT1=14 (1000 kBaud)\n“	Values of the bus timing registers. The name of the configuration is given in brackets.
„I Bus coupling: HIGH\n“	Auto start mode ON/OFF
„I Autostart: ON\n“	Obsolete. Response MAC-List is used by old CANblue device.
„I MAC-List\n:“	Obsolete, see „I MAC-LIST“
„I MAC count: 0\n“	Content of standard ID filter list.
„I STD filter list\n:“	Values of the bus timing registers. The name of the configuration is given in brackets.
„I CAN Id: 1\n“	
„I CAN Id: 4, RTR bit set\n“	
„I STD filter enabled\n“	
„I EXT filter list: \n“	Content of extended ID filter list.
„I CAN Id: 4, RTR bit set\n“	
„I CAN Id: 7FFFF\n“	
„I EXT filter disabled\n“	
„I MAC-Slave: 001122334455 Can-Bluet.-form.: binary, State: disconnected\n“	Information on a SPP server connection: MAC-Address, format of CAN messages (ASCII ,BINARY, OFF), connection status (connected, disconnected).
„I MAC-Master: C44619F9813A Can-Bluet.-form.: off, State: connected\n“	Information on a SPP client.
„I TX-Buff. timeout: 0\n“	Timeout value of the send buffer.
„I OK: CONFIG SHOW\n“	

## 6.2.2 C CAN\_INFO

C	C CAN_INFO	LF/CR-LF
---	------------	----------

Example: „C CAN\_INFO↵ “

### Parameter:

-

### Description:

Information on the current status of the CAN controller and the software queues. Overrun flags are cleared after the response is send.

Information to the TX queue size and transmitted CAN messages since last connection was made are also available. The TX counter is a WORD Value and starts from zero when 65535 by one. With the aid of these values it is possible to implement a TX Handshake. Therefore the difference between the local TX counter and the CANblue TX counter has to be calculated.

## Extended ASCII-Protocol Commands

---

### Response:

„I CAN started\n“ or „I CAN stopped\n“  
„I Tx queue size: 512“  
„I Tx counter: 0“  
„I CAN controller in WARNING LEVEL\n“ or  
„I CAN controller in BUS OFF\n“  
„I Rx CAN controller OVERRUN\n“  
„I Rx SW queue OVERRUN\n“  
„I Tx SW queue OVERRUN\n“  
„I Tx pending“  
„I OK: CAN\_INFO\n“

### Errors:

-

## 6.2.3 C CAN\_INIT

C	CAN_INIT	BAUDRATE	BUSCOP	LF/CR-LF
---	----------	----------	--------	----------

Example: „C CAN\_INIT 500 HIGH↵ “

### Parameter:

Name	Value	Description
BAUDRATE	10-1000 decimal (only CIA standard)	Baud rate in kBaud. CAN controller is initialized with the given baud rate.
BUSCOP	HIGH/LOW (Optional, Default: HIGH)	Mode of bus coupling. Only HIGH is supported.

### Description:

Initializes the CAN controller with the given baud rate. Only CIA standard baud rates are supported:

Standard baud rates according to CIA: 10, 20, 50, 100, 125, 250, 500, 800, 1000 kBaud.

### Response:

„I OK: CAN\_INIT\n“

### Errors:

Error Response	Description
„E 22 Baudrate not supported\n“	Baud rate is not supported. Use one of the CIA baud rate.
„E 31 Error while initializing CAN\n“	Internal error, while initializing CAN controller.
„E 4 Unsupported parameter\n“	Bus coupling „LOW“ is not supported.

## 6.2.4 C CAN\_INIT\_AUTO

C	CAN_INIT_AUTO	TIMEOUT	BUSCOP	LF/CR-LF
---	---------------	---------	--------	----------

Example: „C CAN\_INIT\_AUTO 10 HIGH↵“

Parameter:

Name	Value	Description
TIMEOUT	1-1000 decimal (Optional, Default: 1)	Time in seconds to test for one CIA baud rate.
BUSCOP	HIGH/LOW (Optional, Default: HIGH)	Mode of the bus coupling. Only HIGH is supported.

Description:

CAN controller initialization with automatic baud rate detection. The CAN controller is set into tx-passive mode and all of the CIA baud rates are tested until a valid CAN message is received. The timeout parameter gives the time in seconds for testing for a certain baud rate. The maximum response time is 10 times of the timeout value. If a valid CAN messages is received, the CAN controller is initialized with the found CIA baud rate and a response with this baud rate is sent.

Response:

„I 100\n“ - recognized baud rate 100kBaud  
 „I OK: CAN\_INIT\_AUTO\n“

Errors:

Error Response	Description
„E 23 Baudrate not detected\n“	No valid baud rate could be detected within the given timeout.
„E 4 Unsupported parameter\n“	Bus coupling „LOW“ is not supported.

## 6.2.5 C CAN\_INIT\_CUSTOM

C	CAN_INIT_CUSTOM	BTO	BT1	BUSCOP	NAME	LF/CR-LF
---	-----------------	-----	-----	--------	------	----------

Example: „C CAN\_INIT\_CUSTOM 0 1C HIGH „1000KBAUD CUSTOM”↵“

## Extended ASCII-Protocol Commands

### Parameter:

Name	Value	Description
BT0	0-FF hexadecimal	SJA1000, bit timing register 0
BT1	0-FF hexadecimal	SJA1000, bit timing register 1
BUSCOP	HIGH/LOW (Optional, Default: HIGH)	Mode of the bus coupling. Only HIGH is supported.
NAME	String enclosed in "" Max. 30 characters (Optional)	Name is used for the command „C CONFIG SHOW“.

### Description:

CAN controller initialization with a custom baud rate. The parameters BT0 and BT1 are equivalent to the bus timing register of the CAN controller Philips SJA1000, running at 16 MHz. The bus timing parameter is recalculated internally for the CANblue II CAN controller. The CANblue II CAN controller does not support different sample rates and therefore bit 7 of the parameter BT1 is ignored.

With the parameter "NAME", a name is assigned to the bus timing configuration. The name is used for the command „C CONFIG SHOW\n“. If no name is given, then the baud rate is used as a name.

### Response:

„I OK: CAN\_INIT\_CUSTOM\n“

### Errors:

Error Response	Description
„E 31 Error while initializing CAN\n“	Internal error, while initializing CAN controller.
„E 4 Unsupported parameter\n“	Bus coupling „LOW“ is not supported.

## 6.2.6 C CAN\_START

C	CAN_START	LF/CR-LF
---	-----------	----------

Example: „C CAN\_START↵ “

### Parameter:

-

### Description:

Starts the CAN controller. The message format for transmitting CAN messages over Bluetooth is set to ASCII mode.

### Response:

„I OK: CAN\_START\n“



Errors:

Error Response	Description
„E 32 Error starting CAN\n“	Internal error, while initializing CAN controller.

**6.2.7 C CAN\_STOP**

C	CAN_STOP	LF/CR-LF
---	----------	----------

Example: „C CAN\_STOP↵ “

Parameter:

-

Description:

Stop CAN controller.

Response:

„I OK: CAN\_STOP\n“

Errors:

Error Response	Description
„E 33 Error stop CAN\n“	Internal error, while initializing CAN controller.

**6.2.8 C CAN\_RESET**

C	CAN_RESET	LF/CR-LF
---	-----------	----------

Example: „C CAN\_RESET↵ “

Parameter:

-

Description:

Reset CAN controller.

Response:

„I OK: CAN\_RESET\n“

Errors:

-

### 6.2.9 C AUTOSTART

C	AUTOSTART	MODE	LF/CR-LF
---	-----------	------	----------

Example: „C AUTOSTART ON↵ “

Parameter:

Name	Value	Description
MODE	ON/OFF	Enable or disable auto start mode.

Description:

This command enables or disables auto start mode.  
(see 4.6 Autostart and handshake)

Response:

„I AUTOSTART ON\n“ or „I AUTOSTART OFF\n“  
„I OK: AUTOSTART\n“

Errors:

-

### 6.2.10 C SEND\_CAN\_FRAMES

C	SEND_CAN_FRAMES	MODE	LF/CR-LF
---	-----------------	------	----------

Example: „C SEND\_CAN\_FRAMES ASCII↵ “

Parameter:

Name	Value	Description
MODE	ASCII/BINARY/OFF	Sets the message format for transmitting over Bluetooth.

Description:

Enables or disables transmission of CAN messages in the direction, where the command comes from and sets the message format.

Response:

„I OK: SEND\_CAN\_FRAMES\n“

Errors:

-

## 6.2.11 C FILTER\_ADD

C	FILTER_ADD	MSG_TYP	ID	RTR	LF/CR-LF
---	------------	---------	----	-----	----------

Example: „C FILTER\_ADD STD 3A RTR↵ “

Parameter:

Name	Value	Description
MSG_TYP	STD/EXT	Message type of filter entry (Standard or Extended).
ID	Standard: 0-7FF Extended: 0-1FFFFFFF	CAN ID of the filter entry.
RTR	DATA/RTR (Optional, Default: DATA)	Data or remote frame

Description:

Adds a filter entry to the filter list. If the filter is enabled, only messages which are in the filter list are forwarded. Messages received via Bluetooth are not filtered.

In the filter list, there is space for 4096 standard filter elements.

For extended filter elements there are 300 bytes available, where one element needs 8, 16, 24 or 32 bit depending on the amount of characters of the CAN ID.

CAN-Id Range	used memory in byte
0-7F	1
80-7FFF	2
8000-7FFFFFFF	3
800000-1FFFFFFF	4

Response:

„I OK: FILTER\_ADD↵\n

Errors:

Error Response	Description
„E 41 Error adding ID to filter“	Out of memory for extended filter elements.

### 6.2.12 C FILTER\_REMOVE

C	FILTER_REMOVE	MSG_TYP	ID	RTR	LF/CR-LF
---	---------------	---------	----	-----	----------

Example: „C FILTER\_REMOVE STD 3A RTR↵ “

Parameter:

Name	Value	Description
MSG_TYP	STD/EXT	Message type of filter entry (Standard or Extended).
ID	Standard: 0-7FF Extended: 0-1FFFFFFF	CAN ID of the filter entry.
RTR	DATA/RTR (Optional, Default: DATA)	Data or remote frame

Description:

Removes a filter entry from the filter list.

Response:

„I OK: FILTER\_REMOVE\n“

Errors:

-

### 6.2.13 C FILTER\_CLEAR

C	FILTER_CLEAR	ID_TYP	LF/CR-LF
---	--------------	--------	----------

Example: „C FILTER\_CLEAR EXT↵ “

Parameter:

Name	Value	Description
ID_TYP	STD/EXT	Message type

Description:

Erases the standard or extended filter list.

Response:

„I OK: FILTER\_CLEAR\n“

Errors:

-

### 6.2.14 C FILTER\_ENABLE

C	FILTER_ENABLE	ID_TYP	LF/CR-LF
---	---------------	--------	----------

Example: „C FILTER\_ENABLE EXT↵ “

Parameter:

Name	Value	Description
ID_TYP	STD/EXT	Message type

Description:

Enables the standard or extended filter list. Messages are forwarded, if the ID is found in the filter list. The filter list for standard and extended IDs must be enabled or disabled separately.

Response:

„I OK: FILTER\_ENABLE\n“

Errors:

-

### 6.2.15 C FILTER\_DISABLE

C	FILTER_DISABLE	ID_TYP	LF/CR-LF
---	----------------	--------	----------

Example: „C FILTER\_DISABLE EXT↵ “

Parameter:

Name	Value	Description
ID_TYP	STD/EXT	Message type

Description:

Disables the standard or extended filter list. The filter list for standard and extended IDs must be disabled separately.

Response:

„I OK: FILTER\_DISABLE\n“

Errors:

-

## 6.3 CAN Bluetooth Messages

For transmission of CAN messages over a Bluetooth SPP connection are two frame types (M and X) defined.

## Extended ASCII-Protocol Commands

### 6.3.1 M (ASCII)

M-Type messages are coded in a readable ASCII format.

M	FTD	ID	D0	D1	D2	D3	D4	D5	D6	D7	LF/CR-LF
---	-----	----	----	----	----	----	----	----	----	----	----------

Example: „M SD4 1A2 11 22 33 4↵“

Parameter:

Name	Value	Description
FTD	1.Character - frame format: ,S' = Standard ,E' = Extended 2. Character - frame type: ,D' = Data ,R' = Remote) 3. Character - DLC: ,0-8' Data length	Three characters define the message format.
ID	Standard: 0-7FF hexadecimal Extended: 0-1FFFFFFF hexadecimal	CAN message identifier
D0-D7	0-FF hexadecimal	Message consists of up to 8 data bytes. Every byte is separated by a blank.

Description:

With this command, CAN messages can be send over a Bluetooth connection to another CANblue II. The receiving CANblue II forwards the message to all established Bluetooth connections and if the local CAN controller is started, the message is also send on the CAN network.

Note: Remote messages are send without any data bytes. Nevertheless, the value of the data length (DLC) can be a value between 0 and 8.

Response:

-

Errors:

Error Response	Description
„E 85 Tx SW queue OVERRUN“	Overrun of the send queue, e.g. CAN controller is in error warning or bus off state or the data could not be send fast enough due to slow baud rate.

### 6.3.2 X (Binary)

The binary format allows faster transmission of CAN messages. The format is not in readable ASCII format. The data of the CAN message is send uncoded in a binary value and the fields are not separated by blanks and without any CR/LF character.

Standard CAN message:

,X'	FI	ID_HB	ID_LB	D0	D1	D2	D3	D4	D5	D6	D7
-----	----	-------	-------	----	----	----	----	----	----	----	----

Extended CAN message:

,X'	FI	ID_ HW_HB	ID_ HW_LB	ID_ LW_HB	ID_ LW_LB	D0	D1	D2	D3	D4	D5	D6	D7
-----	----	--------------	--------------	--------------	--------------	----	----	----	----	----	----	----	----

FI bit field

Bit	7	6	5-4	3-0
Type	FF	RTR	0	DLC

Example:

0x58, 0x85, 0x01, 0x02, 0x03, 0x04, 0x19, 0x2A, 0x3B, 0x4C, 0x5D

0x58 → 'X'(binary message type)

0x85 → [FF=1 (Ext); RTR=0 (Data); DLC = 5]

0x01020304 → ID

0x19, 0x2A, 0x3B, 0x4C, 0x5D → 5 data bytes

Parameter:

Name		Value	Description
FI (bit field)	FF(bit7)	0 = Std, 1 = Ext	Frame format Standard or Extended.
	RTR(bit6):	0 = Dat, 1 = RTR	Frame type: Data or Remote
	DLC(bit3-0):	0-8	DLC, Data length
ID_HB		0-7F	high byte of a standard CAN ID
ID_LB		0-FF	low byte of a standard CAN ID
ID_HW_HB		0-1F	high word, high byte of extended CAN ID
ID_HW_LB		0-FF	high word, low byte of extended CAN ID
ID_LW_HB		0-FF	low word, high byte of extended CAN ID
ID_LW_LB		0-FF	low word, low byte of extended CAN ID
D0 – D7		0-FF	up to 8 data bytes

Description:

With this command, CAN messages can be send over a Bluetooth connection to another CANblue II. The receiving CANblue II forwards the message to all established Bluetooth connections and if the local CAN controller is started, the message is also send on the CAN network.

Response:

-

## Extended ASCII-Protocol Commands

---

### Errors:

Error Response	Description
„E 85 Tx SW queue OVER-RUN“	Overrun of the send queue, e.g. CAN controller is in error warning or bus off state or the data could not be send fast enough due to slow baud rate.



## 6.4 Error Response

Error Response	Description
„E 1 Unknown command\n“	An invalid command or message type is received.
„E 2 Wrong parameter\n“	The parameter of a command is invalid..
„E 3 Unsupported command\n“	Received command is not supported (anymore).
„E 4 Unsupported parameter\n“	The parameter of a command is not supported (anymore).
„E 11 Wrong message type\n“	Invalid message type is received. (valid: standard or extended)
„E 12 Wrong frame type\n“	Invalid frame type is received. (valid: data or remote)
„E 13 Wrong data length\n“	Invalid data length is received. (valid: 0-8)
„E 14 Wrong message ID\n“	Invalid ID is received. (valid: 0-7FF or 0-1FFFFFFF)
„E 15 Wrong number of data bytes\n“	Number of data bytes do not match data length.
„E 21 Unknown Bus Coupling value\n“	Invalid bus coupling value. (valid: high)
„E 22 Baudrate not supported\n“	Baud rate is not a CIA baud rate.
„E 23 Baudrate not detected\n“	Automatic baud rate detection could not find a valid baud rate within the given timeout.
„E 31 Error while initializing CAN\n“	CAN controller could not be initialized. Try again to solve the problem.
„E 32 Error starting CAN\n“	CAN controller could not be started. Try again to solve the problem.
„E 33 Error stop CAN\n“	CAN controller could not be stopped. Try again to solve the problem.
„E 41 Error adding ID to filter\n“	Out of memory for extended filter elements.
„E 51 MAC-list is full\n“	No more MAC address could be added. Current firmware supports only one MAC address.
„E 52 Wrong MAC Address\n“	The MAC address is not valid. (valid: 6 byte hexadecimal).
„E 53 MAC Address already exist\n“	The MAC address is already used for a connection to a SPP server.
„E 61 No valid config\n“	There is no valid configuration to load.
„E 63 Error while saving config\n“	Error occurred during saving of the configuration. Configuration is lost.
„E 81 CAN controller in BUS OFF\n“	Error message is send, if the CAN controller is in bus off state.
„E 82 CAN controller in WARNING LEVEL\n“	Error message is send, if the CAN controller is in error warning state.
„E 84 Rx SW queue OVERRUN\n“	One or more consecutive CAN messages are lost due to a software overrun.
„E 85 Tx SW queue OVERRUN\n“	One or more consecutive CAN messages are lost before sending the messages on the CAN bus due to error warning or bus off state of the CAN controller or due to a slow baud rate.
„E 91 Can't show more“	Some filter elements are not shown. Depending on the free space of the send buffer, it is only possible to show a limited amount of filter elements via the command “C CONFIG SHOW”.
„E 99 Unknown Error“	Internal error occurred without a specific error message.

# 7 Appendix

## 7.1 Support

For more information on our products, FAQ lists and installation tips, please refer to the support area on our homepage (<http://www.ixxat.de>). There you will also find information on current product versions and available updates.

## 7.2 Returning hardware

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

## 7.3 Disposing of old equipment

This product is covered by ElektroG (WEEE) and has to be disposed according to ElektroG (WEEE) separately. Products of IXXAT, which are covered by ElektroG, are exclusively for commercial use and marked with the symbol of the crossed-out garbage can.

According to the B2B regulations, the disposal in accordance with § 10 para. 2 clause 3 Electrical and Electronic Equipment act in the version of 16.03.2005 is regulated separately in the General Terms and Conditions and its supplements of IXXAT. The terms and conditions, its supplements and other information on disposal of old equipment can be downloaded at [www.ixxat.de](http://www.ixxat.de).

## 7.4 Information on EMC

The product is a class B device. If the product is used in office or home environment radio interference can occur under certain conditions. To ensure faultless operation of the device, the following instructions must be followed due to technical requirements of EMC:

- use only the included accessories
- the shield of the interfaces must be connected with the device plug and with the plug on the other side

## 7.5 Compliance with RoHS directive

CANBlue II was produced according to the RoHS (Restriction of the use of certain Hazardous substances in electrical and electronic equipment) directive and complies with the directive.

## **7.6 FCC Compliance**

### **Declaration of conformity**

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

- This device may not cause harmful interference, and
- this device must accept any interference received, including interference that may cause undesired operation

FCC Identifier of the built in Bluetooth module:

PVH0939

Test remit:

FCC Rules 47 CFR Part 15 / 2010-01-09

Subpart B - Class B / Section 15.107 and 15.109

in accordance with the procedures given in

ANSI C63.4-2003 – 01/2004

## **7.7 Japan Radio Equipment Compliance (TELEC)**

CANBlue II uses the cB-0939 module which complies with the Japanese Technical Regulation Conformity Certification of Specified Radio Equipment (ordinance of MPT N°. 37, 1981), Article 2, Paragraph 1, Item 19, "2.4GHz band wide band low power data communication system". The cB-0939 MIC certification number is 204WW11100200.



RCB Japan  
Königswinkel 10  
D-32825 Blomberg, Germany  
Phone+49 (0) 52 35 95 00-75  
Fax+49 (0) 52 35 95 00-25  
www.phoenix-testlab.de



# Certificate

No: 204WW11100200

## of Technical Regulations Conformity for Specified Radio Equipment in Japan

PHOENIX TESTLAB GmbH, operating as a Registered Certification Body (RCB ID: 204) with respect to Japan, declares that the listed product complies with the Technical Regulations Conformity Certification of Specified Radio Equipment (ordinance of MPT N°. 37, 1981), Article 2, Paragraph 1, Item 19, "2.4 GHz band wide band low power data communication system"

Product description:	<b>Bluetooth Module</b>
Trademark:	--
Family name:	<b>cB-0939</b>
Type of Designation	cB-OBS433x-04 ; cB-OBS433x-06 ; cB-OBS433i-02 ; cB-OBS433i-04 ; cB-OBS433i-04
Type of Emissions:	<b>78M6F1D</b>
Frequency:	<b>2.402 – 2.480 GHz (79 Ch)</b>
Antenna power:	<b>0.30 mW/MHz</b>
Serial No:	--
Software Release No:	--
Manufacturer:	<b>ONROX AB</b>
Address:	<b>Jägersrovägen 204</b>
City:	<b>Malmö SE-213 77</b>
Country:	<b>Sweden</b>

This certificate is granted to:

Name:	<b>connectBlue AB</b>
Address:	<b>Norra Vallgatan 64 3V</b>
City:	<b>Malmö SE-211 19</b>
Country:	<b>Sweden</b>

This certificate has 2 annexes.

Blomberg, 17 March 2011  
Place, Date

PHOENIX  
**TESTLAB** GMBH  
32825 BLOMBERG  
Signature, Stamp

## 7.8 EC Declaration of Conformity

IXXAT Automation hereby declares  
that the product:

with the article numbers:

	1.01.0126.11000
	1.01.0126.11001
	1.01.0126.12000
	1.01.0126.12001


do comply with the EC directives 2004/108/EC und 1999/5/EC Art. 3b.

Applied harmonized standards in particular:

ETSI EN 301489-1 / V1.8.1  
(2008-04)

EN 61000-6-2:2005

11.10.2011, Dipl.-Ing. Christian Schlegel , Managing Director



IXXAT Automation GmbH  
Leibnizstr. 15  
88250 Weingarten

### 7.9 Technical Specifications

Microcontroller:	ST-Microelectronics STM32F103RC / 72 MHz
Bluetooth radio:	ST-Ericsson STLC2500DB
Bluetooth RF output power:	Class 1, max 14 dBm (conducted - excluding antenna gain)
Bluetooth receive sensitive level:	-91 dBm
Bluetooth input level (max):	+5 dBm
Bluetooth output frequency:	2.402 - 2.480 GHz, ISM band
Bluetooth stack:	connectBlue Embedded Bluetooth Stack
Bluetooth qualification:	2.1 + EDR
CAN transceiver:	Texas Instruments SN65HVD251
Max. number of CAN bus nodes:	120
Power supply:	9 - 30 V DC
Power consumption:	typically 50 mA at 12 V
Dimensions (L x W x H) in mm:	81 x 66 x 26
Weight:	approx. 83 g
Working temperature range:	-40°C to +85°C
Relative humidity:	10-95 %, non-condensing
Protection type:	IP 20
CAN interface isolation working voltage:	130 V AC/DC (Continuous) 1000 V DC (1 Second)
External antenna version:	RP-SMA connector, max. antenna gain 3.4 dBi
Bridge set-up time:	typically 3-4 seconds
Bluetooth transfer delay:	approx. 4 ms (average) (CAN-Bluetooth or Bluetooth-CAN) *
CAN transmission rate:	100% Bus load at 1 MBit
Maximal distance between two devices in bridge mode:	300 meter / 1000 feet (internal antenna version)

---

\* Depending on configured Bluetooth connection profile („D LINK\_POLICY SHORTEST\_LATENCY“), connection quality and distance between sender and receiver.