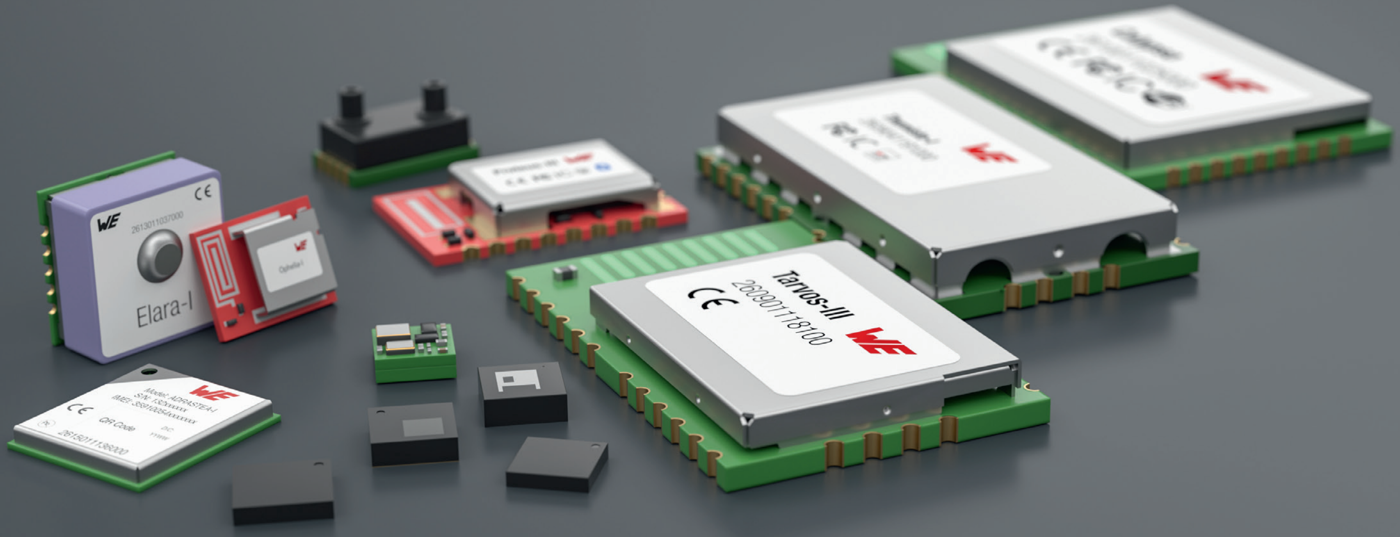


ANR004

Proteus How to use the peripheral only mode

Version 2.6

September 9, 2024

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT

Revision history

Manual version	Notes	Date
1.0	• Initial version	February 2017
1.1	• Updated MTU size to 247 bytes	July 2017
2.0	• New corporate design	June 2018
2.1	• Updated product name from AMB2621 to Proteus-I	November 2018
2.2	• Updated file name to new AppNote name structure. Updated important notes, legal notice & license terms chapters.	June 2019
2.3	• Added Proteus-II and Proteus-III description • Updated address of Division Wireless Connectivity & Sensors location	January 2020
2.4	• Restructured app note • Added new chapter `Quickstart` with new connection setup examples • Added information on the Proteus-III mini EV-Board	February 2021
2.5	• Updated `Important notes`, meta data and document style	July 2023

2.6	• Corrected figure 3 • Proteus Connect app has new name "WE Bluetooth LE Terminal" app	September 2024

Abbreviations

Abbreviation	Name	Description
BTMAC		Bluetooth® conform MAC address of the module used on the RF-interface.
CS	Checksum	Byte wise XOR combination of the preceding fields.
DTM	Direct test mode	Mode to test Bluetooth® specific RF settings.
GAP	Generic Access Profile	The GAP provides a basic level of functionality that all Bluetooth® devices must implement.
I/O	Input/output	Pinout description.
LPM	Low power mode	Mode for efficient power consumption.
LSB	Least significant bit	
MAC		MAC address of the module.
MSB	Most significant bit	
MTU	Maximum transmission unit	Maximum packet size of the Bluetooth® connection.
Payload		The intended message in a frame / package.
RF	Radio frequency	Describes wireless transmission.
RSSI	Receive Signal Strength Indicator	The RSSI indicates the strength of the RF signal. Its value is always printed in two's complement notation.
Soft device		Operating system used by the nRF52 chip.
UART	Universal Asynchronous Receiver Transmitter	Allows the serial communication with the module.
[HEX] 0xhh	Hexadecimal	All numbers beginning with 0x are hexadecimal numbers. All other numbers are decimal, unless stated otherwise.

Contents

1 Introduction

The Proteus is a Bluetooth® module based on the nRF52 Nordic Semiconductors SoC which provides various Bluetooth® LE and low power features.
In addition to the standard command mode, that uses predefined commands to run and configure the radio module, Würth Elektronik eiSos launches the "peripheral only mode" on the Proteus to use the module as Bluetooth® LE bridge in a simple way.
In this mode, a Bluetooth® LE interface using the static passkey authentication method (with bonding) and a transparent UART interface is provided, such that no configuration of the module is required to equip a custom application with it.
In case the user needs a non-standard configuration, it can be configured in advance using the command mode, or upon request Würth Elektronik eiSos can apply customer specific configurations during the production process.
The following chapters describe how to set the module into peripheral only mode and which steps have to be applied to establish a connection to the radio module.

2 Prerequisites

- A Proteus EV-Board in factory state, for example
  - the Proteus-I EV-Board with firmware version 3.0.0 or newer.
  - the Proteus-II EV-Board.
  - the Proteus-III EV-Board or mini EV-Board.
- A central device, that initiates the connetion setup. For example
  - a smart phone with Bluetooth® LE function and the Nordic Semiconductor nRF Connect App.
  - another Proteus EV-Board or mini EV-Board.
  - a Proteus USB radio stick.

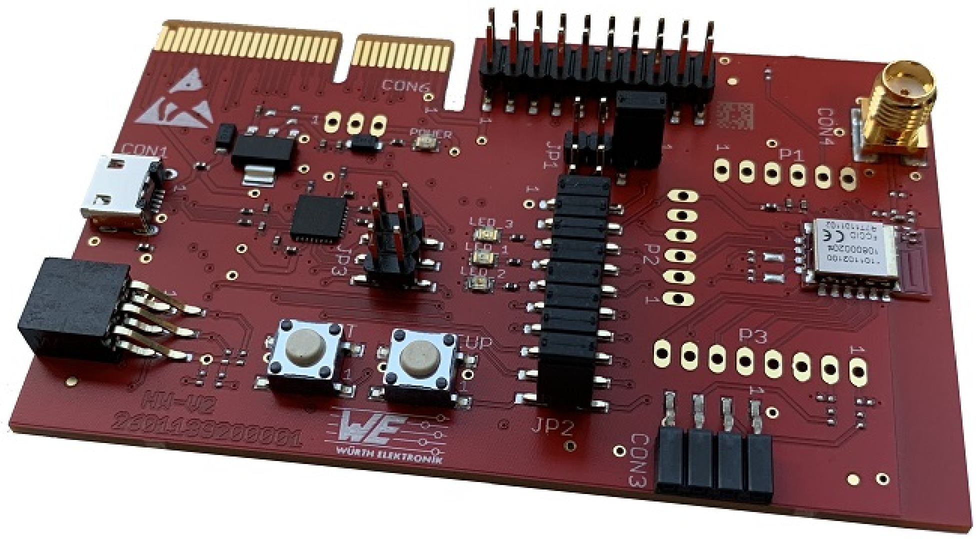


Figure 1: Proteus-III EV-Board

To be sure that the Proteus radio module or Proteus USB radio stick is in factory state, please run a factory reset before doing any other action.

Please check whether the most recent firmware is installed on any Proteus radio module, EV-Board or Proteus USB radio stick.

3 Peripheral only mode: General information

For a better understanding of the content of this chapter, basic knowledge of the Bluetooth® standard as well as that of the SPP-like profile is of advantage. Please find more details on that in the respective advanced developer guide:

- ANR002 Proteus-I advanced developer guide [1]
- ANR005 Proteus-II advanced developer guide [2]
- ANR009 Proteus-III advanced developer guide [3]

3.1 How to set the Proteus radio module to peripheral only mode?

The Proteus starts in peripheral only mode, when a HIGH level is applied at the *OPERATION_MODE* pin and a reset is done via the */RESET* pin. If the *OPERATION_MODE* pin is LOW during the reset, the module starts in normal operation mode with command interface.

A pull-down is applied to the *OPERATION_MODE* pin during start-up. Thus increased currents can occur for a period $\leq$ 1 ms.

After the start-up procedure has been finished, the *OPERATION_MODE* pin and thus the applied signal level has no function.

For Proteus-III, the *OPERATION_MODE* pin has been renamed to *MODE_1*, while maintaining the same function. Throughout this app note we will use *OPERATION_MODE* as a term for this pin.

In case of the EV-Board for Proteus, simply connect the *OPERATION_MODE* pin to *VCC* by setting the respective jumper (see figure 2, 3 and 4). Then press the reset button to start the module in peripheral only mode.

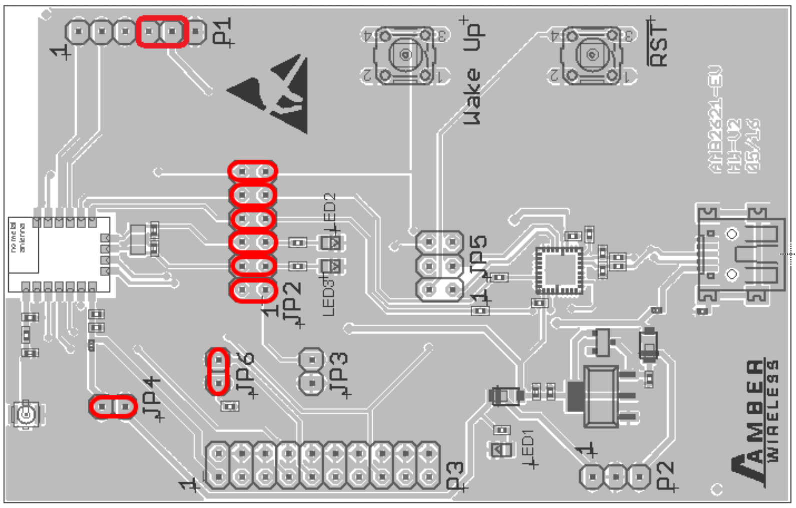


Figure 2: On Proteus-I and Proteus-II EV-Board, set these jumpers to start the peripheral only mode after reset.

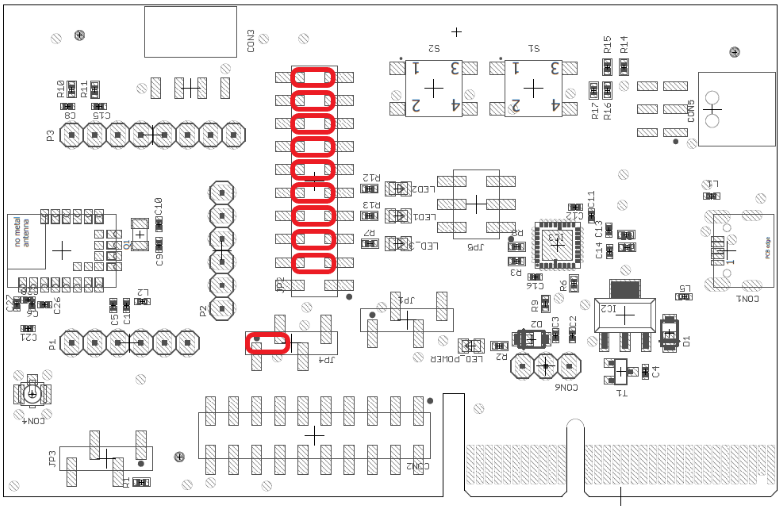


Figure 3: On Proteus-III EV-Board, set these jumpers to start the peripheral only mode after reset.

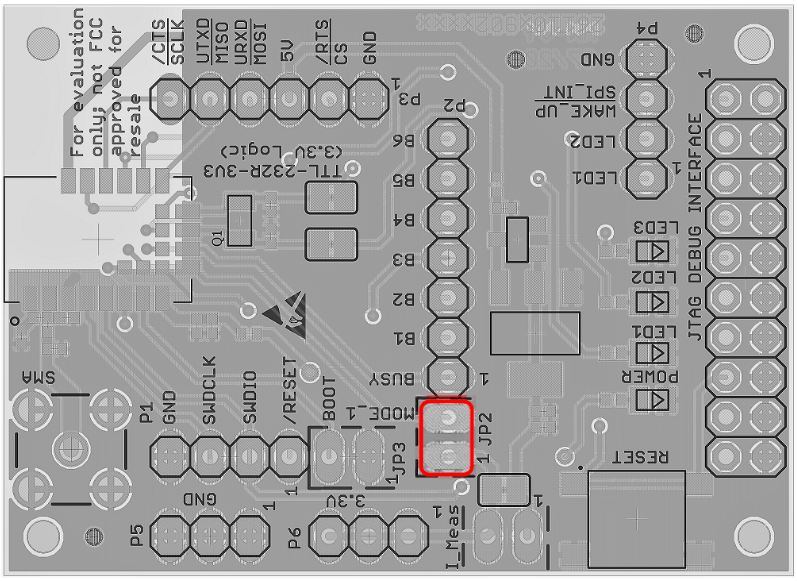


Figure 4: On Proteus-III mini EV-Board, set these jumpers to start the peripheral only mode after reset.

3.2 General connection setup information

In factory state, the peripheral only mode uses the static passkey pairing with bonding authentication method, which requests a static passkey from the connecting device. Figure 5 shows the steps that have to be performed successively during connection setup using the static passkey pairing method:

1. Physical connection establishment
   A physical connection has to be established first. Therefore, a central device (usually smart phone) has to connect to the Proteus which runs as peripheral.

2. Pairing process
   The authentication and exchange of encryption information is part of the pairing process. The central device must request at least the same security level to access the characteristics of the Proteus. The peripheral only mode uses static passkey bonding by default. The Proteus waits for the bonding request of the central device to perform this step.

STOP In case the central device goes on with the next steps without placing this bonding request, the peripheral device disconnects immediately as the required security level is not achieved. The same holds, if the central device places a bonding request with lower security level than required by the peripheral device (static passkey with bonding).

3. Exchange of the maximum transmission unit (MTU)
   The maximum transmission unit can be increased to allow the transmission of larger data packets. The Proteus allows an MTU of up to 247 bytes, which results in a payload of up to 243 bytes. This step is optional. Not selecting a higher MTU will use the Bluetooth® LE 4.0 default MTU which results in 19 bytes payload for the user but will be compatible to pre Bluetooth® LE 4.2 devices.

4. Discover the characteristics of the Proteus SPP-like profile
   The characteristics offered by the Proteus have to be discovered by the central.

5. Notification enable
   The peripheral must let the central know, when there is new data. Therefore, notifications have to be enabled. After this step, the channel is open and data transmission can start.

For the description, we assume that a smart phone is the initiator of the connection. Thus, it acts as central and the Proteus acts as peripheral in figure 5.

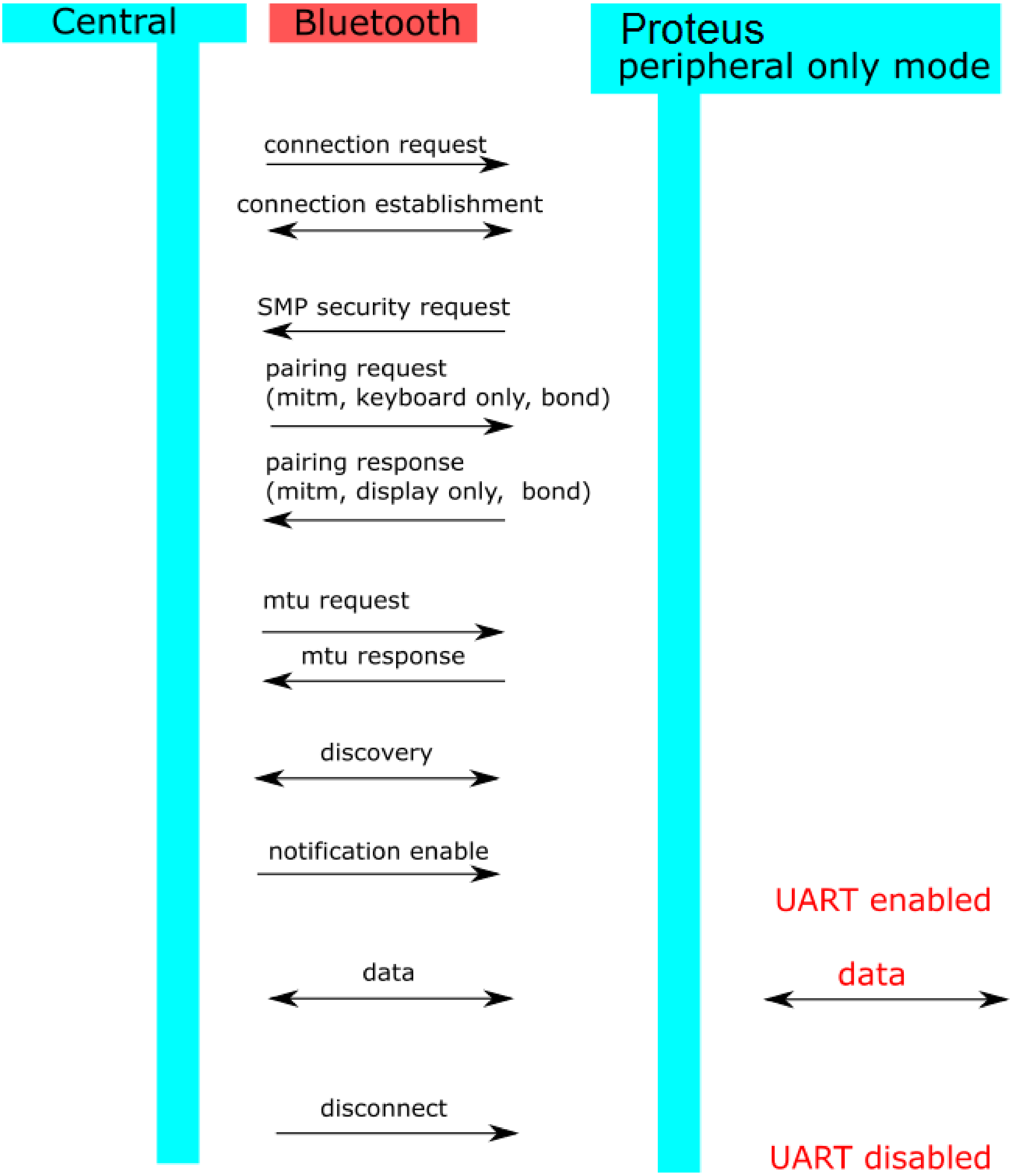


Figure 5: Steps for the connection setup in static passkey mode (default)

3.3 Preconfiguring of the module

In case user settings (such as UART baud rate, security mode or the static passkey value) have to be modified, please start the module in normal mode (apply a low signal at the *OPERATION*

MODE pin during start-up). Then use the commands like `CMD_SET_REQ` to update these user settings and switch back to peripheral only mode (apply a high signal to the *OPERATION MODE* pin during start-up).

STOP For security reasons it is strongly recommended to change the default `RF_StaticPasskey` to a customer specific passkey.

Custom product: Upon request Würth Elektronik eiSos can apply customer specific configuration(s) during the production process.

4 Quickstart

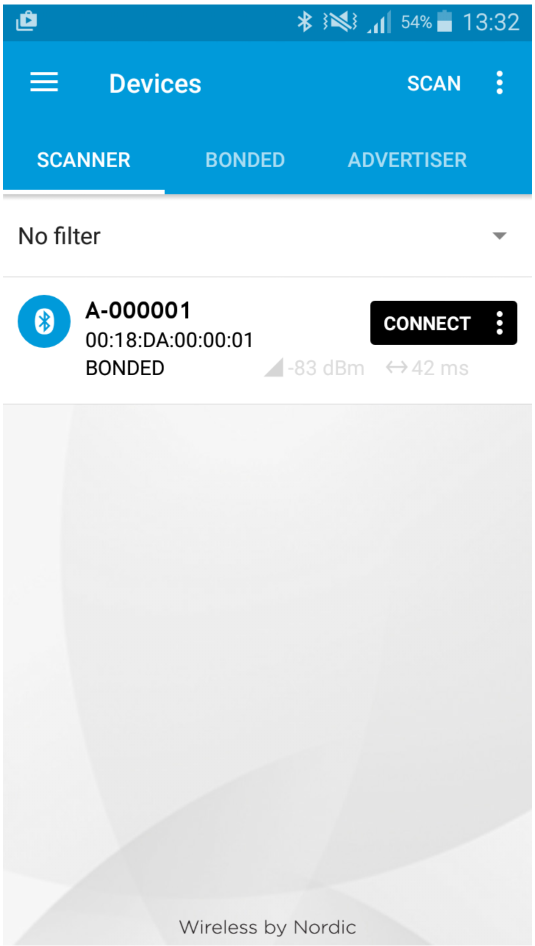
In chapter 3.2, it has been described which steps have to be performed by the central device to setup a connection to a Proteus radio module running in peripheral only mode. What this means in practice will be shown in this chapter. Two examples are following. First, how to use a smart phone and the nRF Connect App to setup a connection to a Proteus radio module running in peripheral only mode (see chapter 4.1). And second, how to use another Proteus radio module or Proteus USB radio stick to do so (see chapter 4.3).

4.1 Smart phone using nRFConnect app as central device

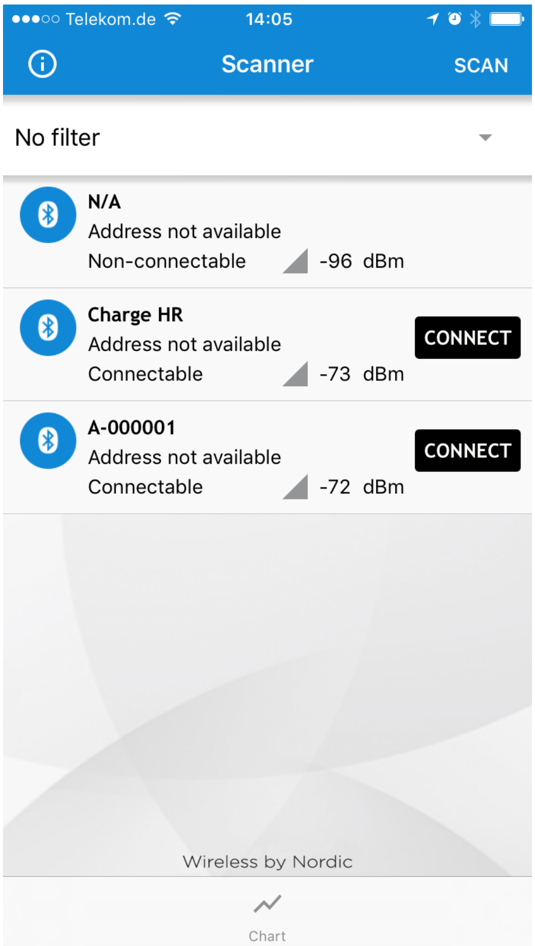
This chapter describes how to setup a connection to the Proteus radio module in peripheral mode (factory state), when a smart phone and the nRF Connect App are used.

! The nRF Connect App is an open source App providing standard Bluetooth® LE functions for iOS as well as for Android devices.

Please perform the following steps:

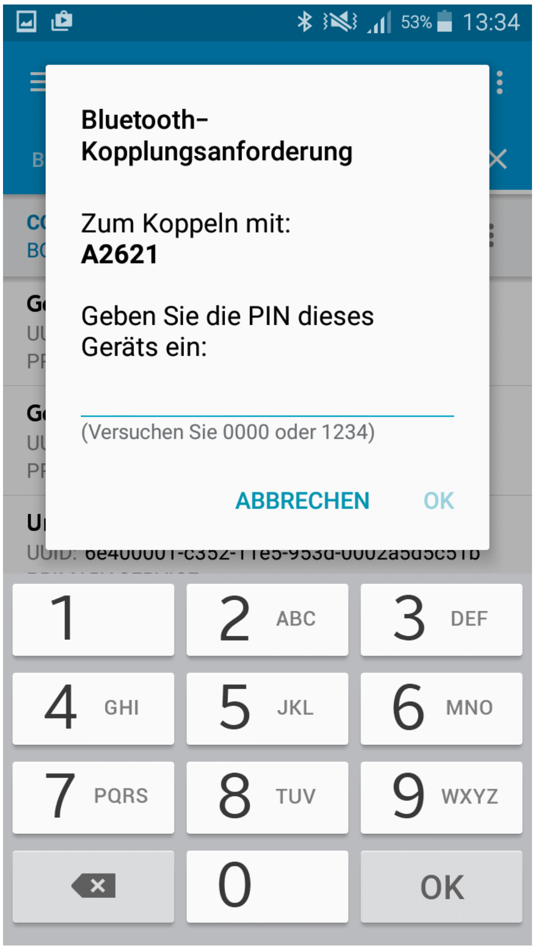
<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td colspan="2">• Connect the module to a PC and open a terminal program using the Proteus default UART settings (115200 Baud, 8n1).
• Set the module into peripheral only mode as described in chapter `3.1`. Initially, the module is advertising. Thus the Proteus *LED_1* is blinking.
• Start your smart phone, enable the Bluetooth® LE feature and start the nRF Connect App.
• Press "SCAN" to find the module on the radio.
• When the module A-xxxxxx appears, press connect. (Note: the part after "A-" is the 3 LSB as ASCII hex of the BTMAC, the fixed part "0x0018DA" is not part of the device descriptor).</td></tr>
<tr><td>

54% 13:32
Devices
SCAN
SCANNER
BONDED
ADVERTISER
No filter
CONNECT
Wireless by Nordic

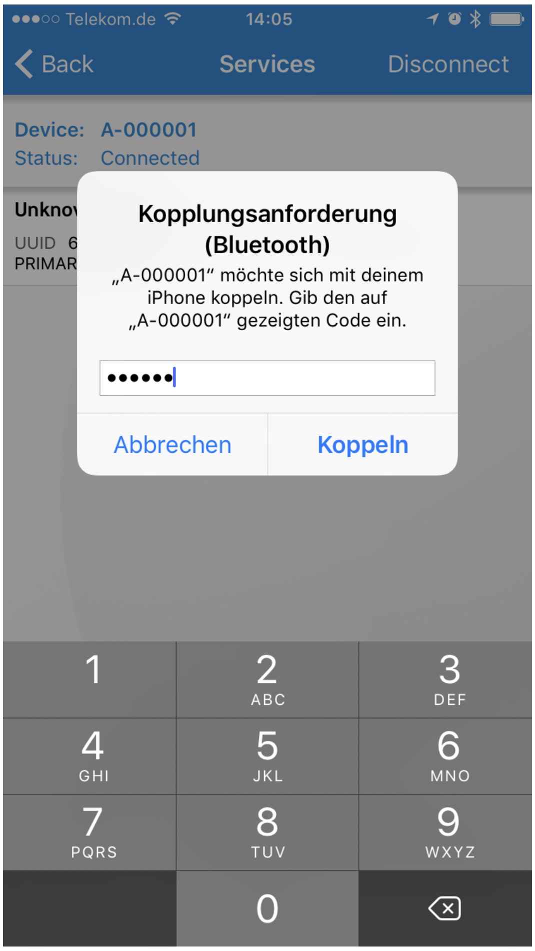
A-000001
00:18:DA:00:00:01
BONDED
-83 dBm
42 ms</td><td>

Telekom.de
14:05
Scanner
SCAN
No filter
CONNECT
CONNECT
Wireless by Nordic
Chart

N/A
Address not available
Non-connectable -96 dBm
Charge HR
Address not available
Connectable -73 dBm
A-000001
Address not available
Connectable -72 dBm</td></tr>
</table>

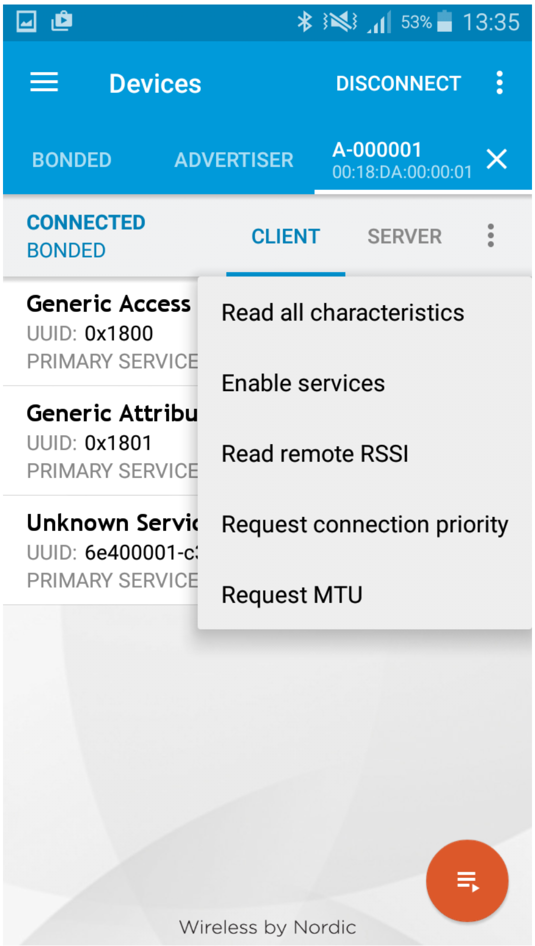
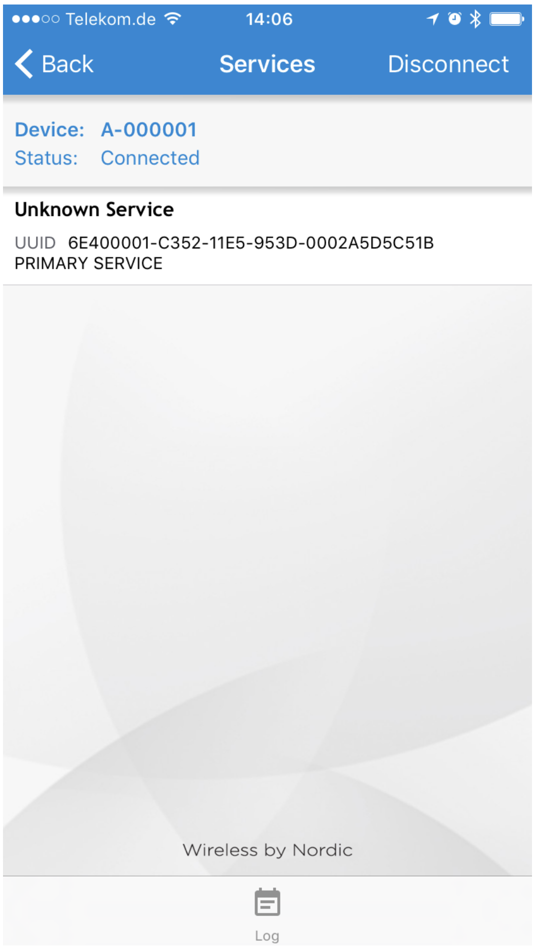
<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td colspan="2">- As soon as the module has received the connection request the module *LED_1* (*LED_3* on the Proteus-EV) will constantly light up.
- Then the radio module requests for the static passkey. In default, the passkey is "123123".
- The Bluetooth® coupling requirement popup is shown in your smart phone.
- When the bonding feature is enabled in the authentication settings and the bonding information already exists, a re-entering of the passkey is not required when reconnecting.</td></tr>
<tr><td>
Bluetooth-Kopplungsanforderung
Zum Koppeln mit:
A2621
Geben Sie die PIN dieses Geräts ein:
(Versuchen Sie 0000 oder 1234)

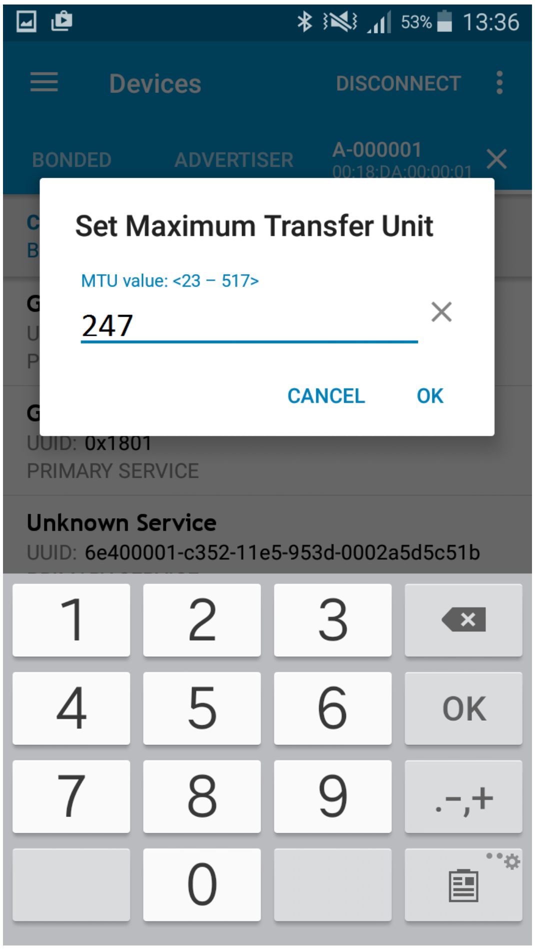
53% 13:34
UUID: 6e400001-c352-11e5-953d-0002a5d5c51b
ABBRECHEN
OK
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
0
OK

</td><td>
Device: A-000001
Status: Connected
Kopplungsanforderung (Bluetooth)
„A-000001“ möchte sich mit deinem iPhone koppeln. Gib den auf „A-000001“ gezeigten Code ein.
••••••

Telekom.de 14:05
Back
Services
Disconnect
Abbrechen
Koppeln
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
0

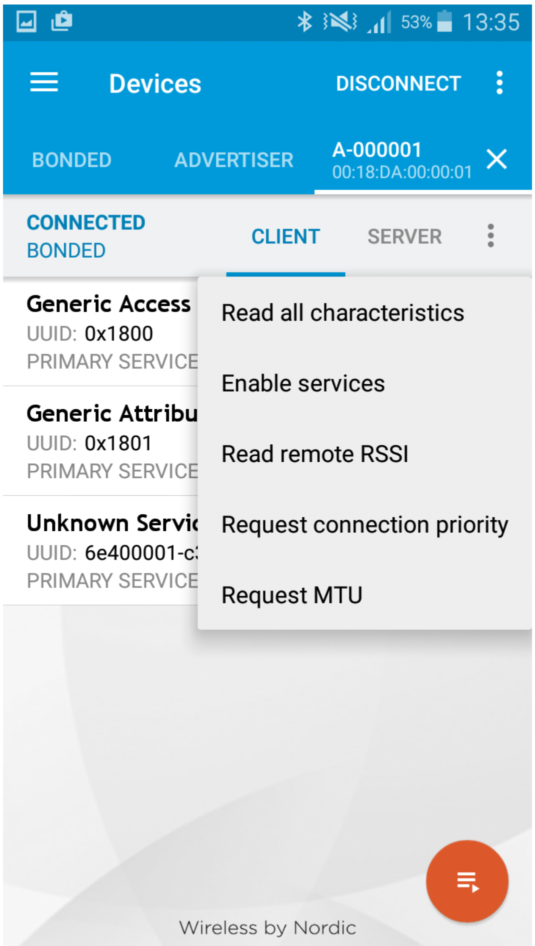
</td></tr>
</table>

Android	iOS
• Now you are authenticated. • Please click on the menu bullets on the right and press "Request MTU" to request for a larger MTU.  53% 13:35 Devices DISCONNECT BONDED ADVERTISER A-000001 00:18:DA:00:00:01 CONNECTED BONDED CLIENT SERVER Generic Access UUID: 0x1800 PRIMARY SERVICE Generic Attribu UUID: 0x1801 PRIMARY SERVICE Unknown Servic UUID: 6e400001-c PRIMARY SERVICE Read all characteristics Enable services Read remote RSSI Request connection priority Request MTU Wireless by Nordic	• Now you are authenticated. • Please click on the "Unknown Service" to start the service discovery and the MTU request.  ●●●○○ Telekom.de 14:06 Back Services Disconnect Device: A-000001 Status: Connected Unknown Service UUID 6E400001-C352-11E5-953D-0002A5D5C51B PRIMARY SERVICE Wireless by Nordic Log

Android	iOS
• The Proteus allows an MTU of up to 247 bytes, which results in a payload size of 243 bytes.  53% 13:36 Devices DISCONNECT BONDED ADVERTISER A-000001 00:18:DA:00:00:01 Set Maximum Transfer Unit MTU value: <23 – 517> 247 CANCEL OK UUID: 0x1801 PRIMARY SERVICE Unknown Service UUID: 6e400001-c352-11e5-953d-0002a5d5c51b	• The iOS App runs this step simultaneously in the background, a user-defined MTU is not possible.

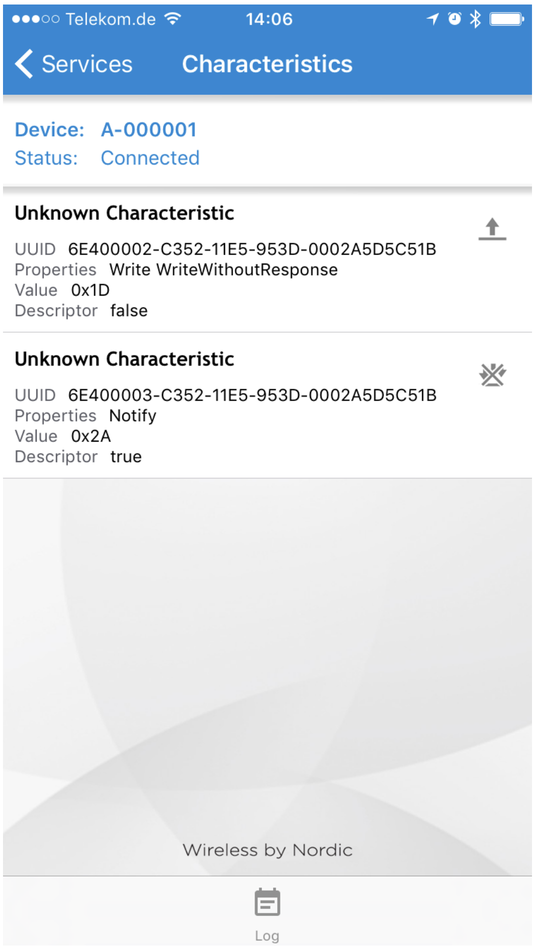
<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td>

- Again click on the menu bullets on the right and press "Enable services" to enable the notifications.



</td><td>

- Press the arrows on the RX-characteristic `6E400003- C352- 11E5- 953D -0002A5D5C51B` to enable the notifications. Press it until a cross appears (see below, it has to be pressed at least once). If a cross is already shown press it twice so the cross disappears and then reappears.

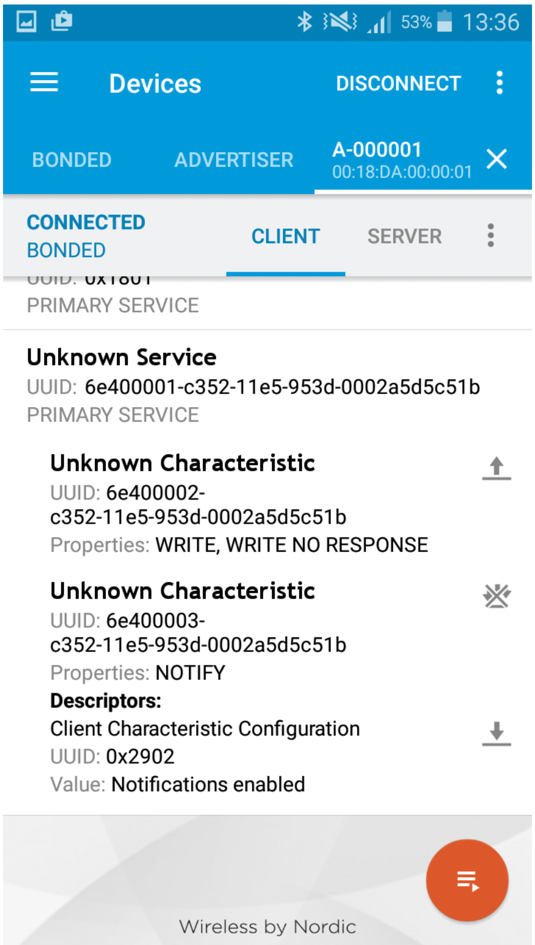
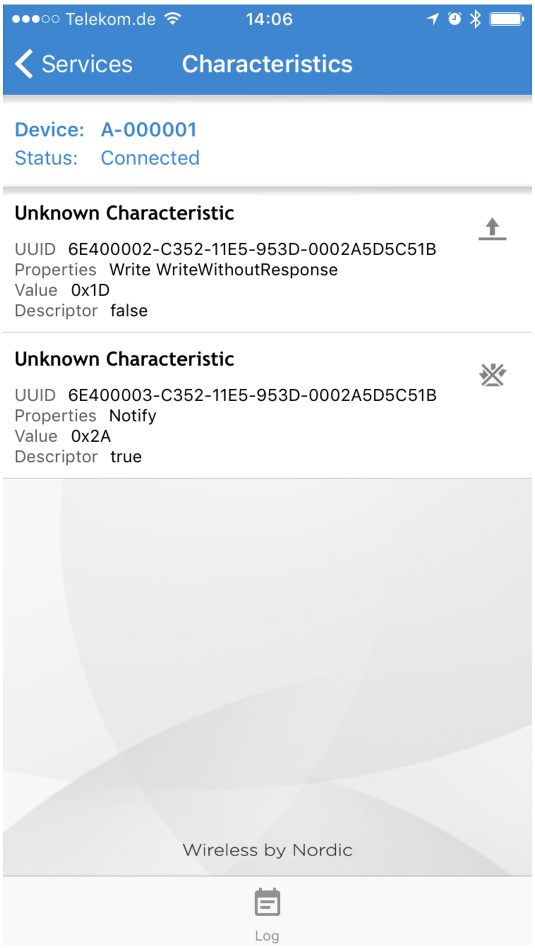


Device: A-000001
Status: Connected

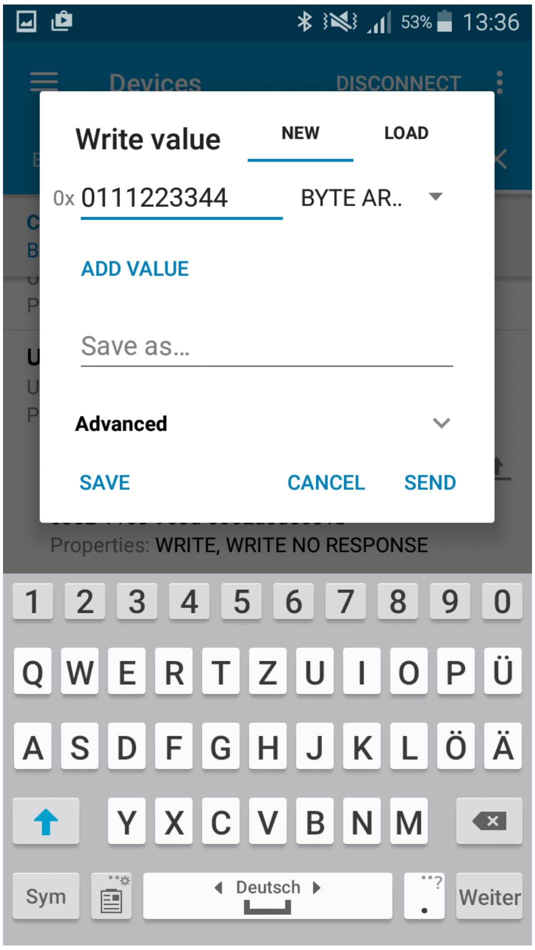
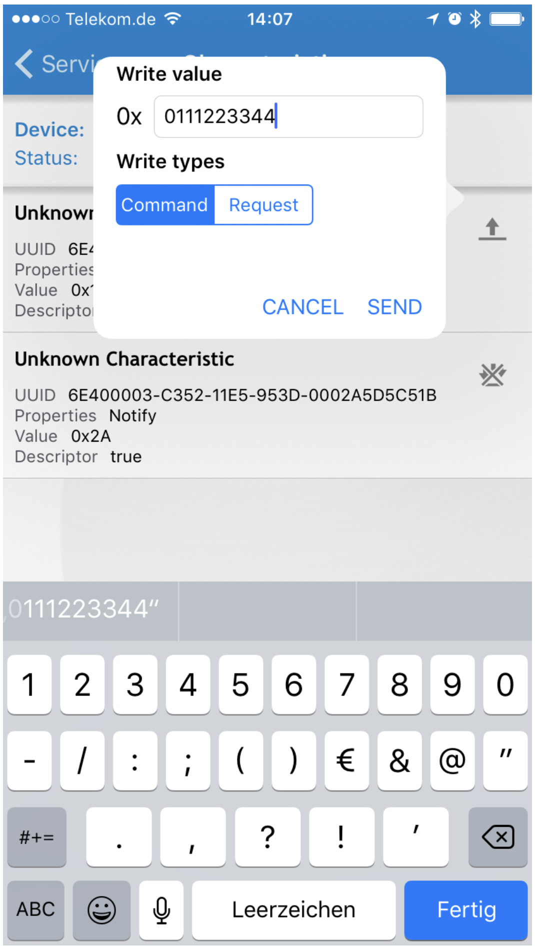
Unknown Characteristic
UUID 6E400002-C352-11E5-953D-0002A5D5C51B
Properties Write WriteWithoutResponse
Value 0x1D
Descriptor false

Unknown Characteristic
UUID 6E400003-C352-11E5-953D-0002A5D5C51B
Properties Notify
Value 0x2A
Descriptor true
</td></tr>
<tr><td colspan="2">

- As soon as the module has received the notification enable request the Proteus *LED_2* (*LED_2* on the Proteus-EV) is turned on.
</td></tr>
</table>

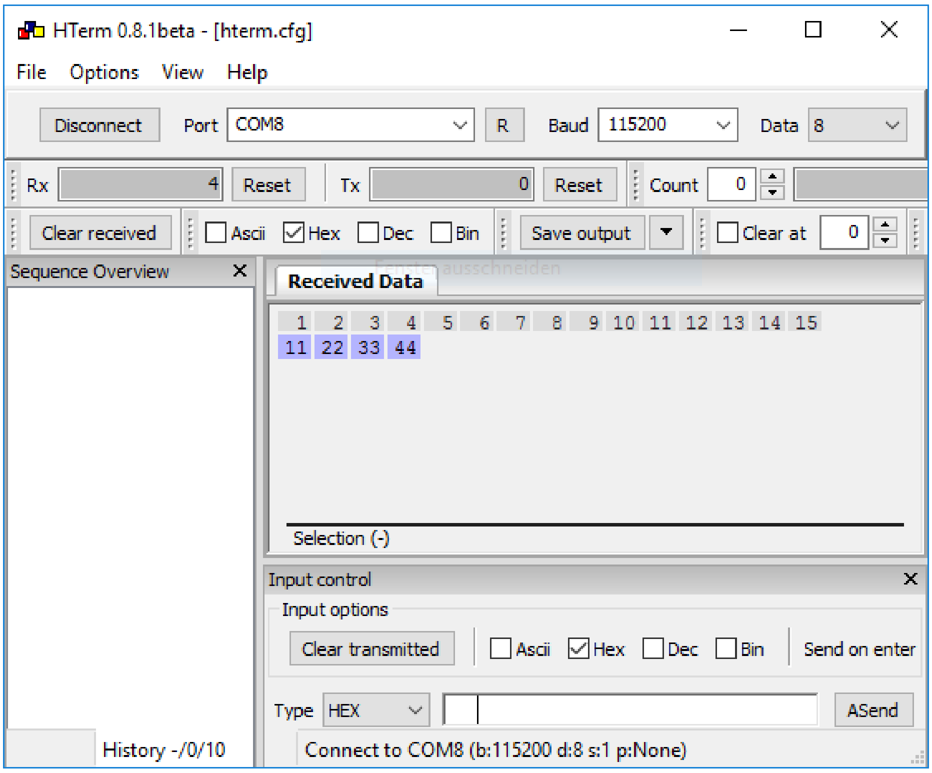
Android	iOS
 53% 13:36 Devices DISCONNECT BONDED ADVERTISER A-000001 00:18:DA:00:00:01 CONNECTED BONDED CLIENT SERVER UUID: 0x1801 PRIMARY SERVICE **Unknown Service** UUID: 6e400001-c352-11e5-953d-0002a5d5c51b PRIMARY SERVICE **Unknown Characteristic** UUID: 6e400002-c352-11e5-953d-0002a5d5c51b Properties: WRITE, WRITE NO RESPONSE **Unknown Characteristic** UUID: 6e400003-c352-11e5-953d-0002a5d5c51b Properties: NOTIFY **Descriptors:** Client Characteristic Configuration UUID: 0x2902 Value: Notifications enabled Wireless by Nordic	 ●●●○○ Telekom.de 14:06 Services Characteristics Device: A-000001 Status: Connected **Unknown Characteristic** UUID 6E400002-C352-11E5-953D-0002A5D5C51B Properties Write WriteWithoutResponse Value 0x1D Descriptor false **Unknown Characteristic** UUID 6E400003-C352-11E5-953D-0002A5D5C51B Properties Notify Value 0x2A Descriptor true Wireless by Nordic Log

- Now you are fully connected and you can access the characteristics. The maximum size of payload depends on the chosen MTU size. Here we chose 247 bytes, which allows us to send 243 bytes of payload via the channel.
- To send data to the Proteus, press the arrow next to the TX-characteristic `6E400002-C352-11E5-953D-0002A5D5C51B`.
- Then enter 0x01 as header byte followed by your payload (for example 0x11 0x22 0x33 0x44) and press "SEND". The payload size is dependent on the MTU that was negotiated in the connection process. The smallest supported MTU for all Bluetooth® 4.0 (or newer) devices results in a max payload (after the 0x01 header) of 19 bytes.

Android	iOS
 Write value 0x 0111223344 BYTE AR.. ADD VALUE Save as... Advanced Properties: WRITE, WRITE NO RESPONSE	 Write value 0x 0111223344 Write types Command Request Unknown Characteristic UUID 6E400003-C352-11E5-953D-0002A5D5C51B Properties Notify Value 0x2A Descriptor true

- The payload that has been sent via radio is output by the Proteus via UART. In peripheral only mode, a transparent UART interface is used. This means, that only payload data is transmitted, without any packet header or footer. Thus the transmitted bytes 0x11 0x22 0x33 0x44 are displayed on the connected terminal program.

Android	iOS


HTerm 0.8.1beta - [hterm.cfg]

Port COM8, Baud 115200, Data 8

Rx 4, Tx 0, Count 0

☐ Ascii ☑ Hex ☐ Dec ☐ Bin

Received Data:

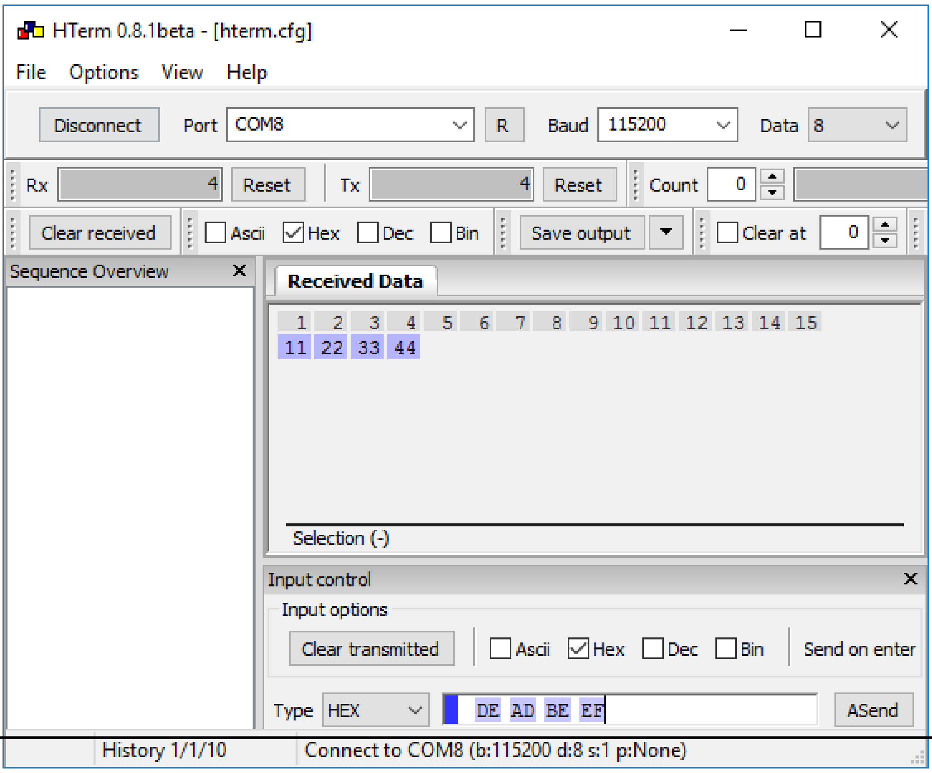
```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
11 22 33 44
```

Input options: ☐ Ascii ☑ Hex ☐ Dec ☐ Bin, Type HEX

History -/0/10

Connect to COM8 (b:115200 d:8 s:1 p:None)

- To send back data simply enter your payload in the respective terminal program field and press enter. In this example we choose 0xDE 0xAD 0xBE 0xEF. The header 0x01 will be automatically applied by the module and is not to be transmitted by the host.
- Here again the maximum payload size (MTU) must be respected.


HTerm 0.8.1beta - [hterm.cfg]

Port COM8, Baud 115200, Data 8

Rx 4, Tx 4, Count 0

☐ Ascii ☑ Hex ☐ Dec ☐ Bin

Received Data:

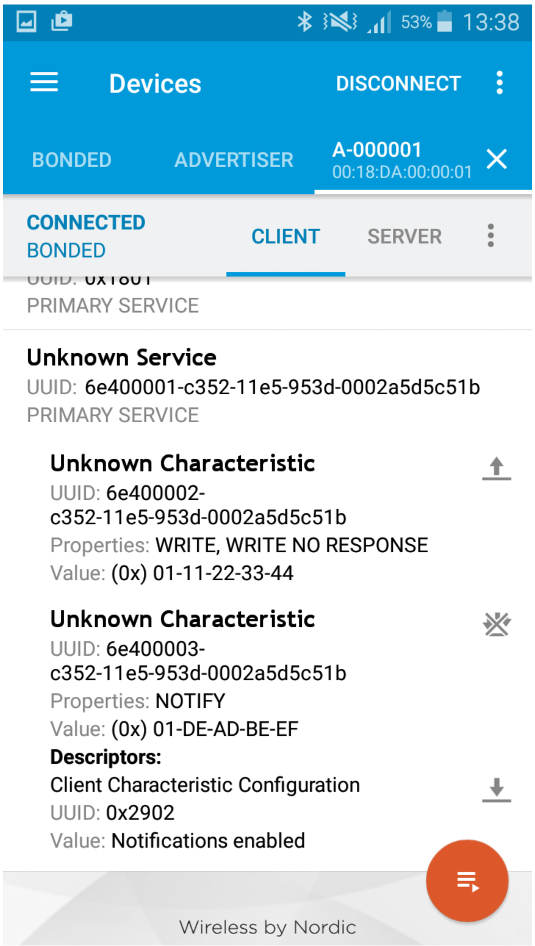
```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
11 22 33 44
```

Input options: ☐ Ascii ☑ Hex ☐ Dec ☐ Bin, Type HEX

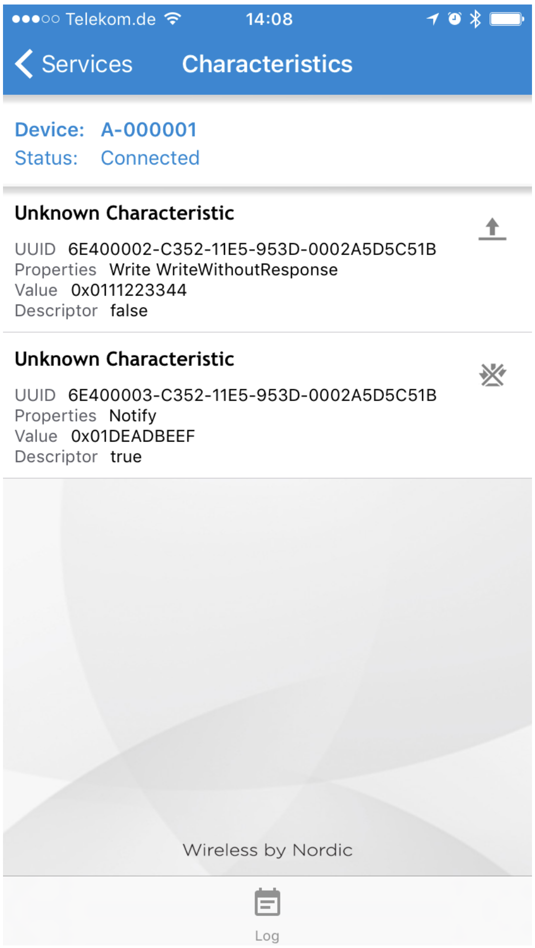
Input: DE AD BE EF

History 1/1/10

Connect to COM8 (b:115200 d:8 s:1 p:None)

<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td colspan="2">• The received data can be found in the RX-characteristic <code>6E400003-C352-11E5-953D-0002A5D5C51B</code>. It contains the header byte 0x01 and the payload 0xDE 0xAD 0xBE 0xEF.</td></tr>
<tr><td>

53% 13:38
Devices
DISCONNECT
BONDED
ADVERTISER
A-000001
00:18:DA:00:00:01
CONNECTED
BONDED
CLIENT
SERVER

UUID: 0x1801
PRIMARY SERVICE
Unknown Service
UUID: 6e400001-c352-11e5-953d-0002a5d5c51b
PRIMARY SERVICE
Unknown Characteristic
UUID: 6e400002-c352-11e5-953d-0002a5d5c51b
Properties: WRITE, WRITE NO RESPONSE
Value: (0x) 01-11-22-33-44
Unknown Characteristic
UUID: 6e400003-c352-11e5-953d-0002a5d5c51b
Properties: NOTIFY
Value: (0x) 01-DE-AD-BE-EF
<b>Descriptors:</b>
Client Characteristic Configuration
UUID: 0x2902
Value: Notifications enabled
 </td>
<td>

●●●○○ Telekom.de
14:08
Services
Characteristics

Device: A-000001
Status: Connected
Unknown Characteristic
UUID 6E400002-C352-11E5-953D-0002A5D5C51B
Properties Write WriteWithoutResponse
Value 0x0111223344
Descriptor false
Unknown Characteristic
UUID 6E400003-C352-11E5-953D-0002A5D5C51B
Properties Notify
Value 0x01DEADBEEF
Descriptor true

Wireless by Nordic
Log

</td></tr>
</table>

4.2 Smart phone using WE Bluetooth LE Terminal app as central device

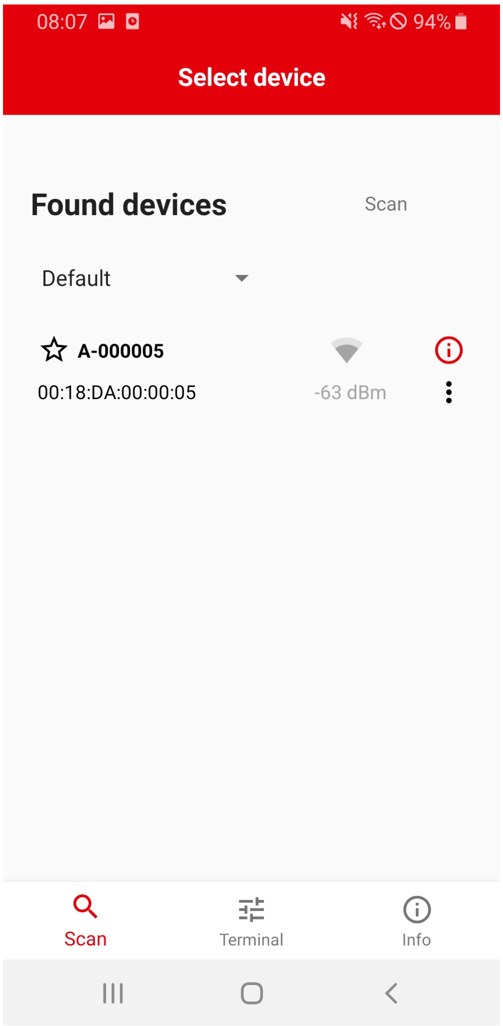
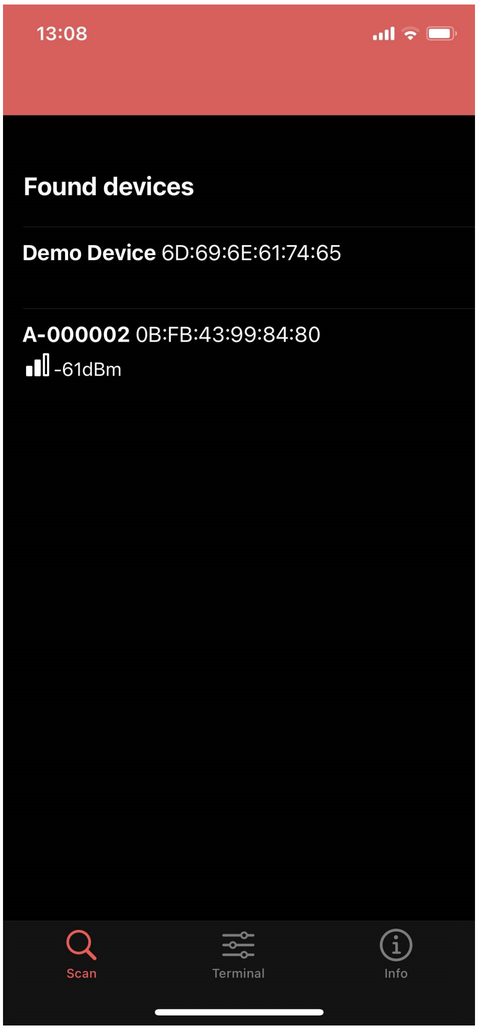
This chapter describes how to setup a connection to the Proteus radio module in peripheral mode (factory state), when a smart phone and the WE Bluetooth LE Terminal App are used.

The WE Bluetooth LE Terminal App for iOS and Android is provided by Würth Elektronik eiSos as executable [4, 5] as well as source code [6].

Please perform the following steps:

<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td colspan="2">• Connect the module to a PC and open a terminal program using the Proteus default UART settings (115200 Baud, 8n1).
• Set the module into peripheral only mode as described in chapter 3.1. Initially, the module is advertising. Thus the Proteus <i>LED_1</i> is blinking.
• Start your smart phone, enable the Bluetooth® LE feature and start the WE Bluetooth LE Terminal App.</td></tr>
</table>

Please note that Bluetooth® LE function of Android devices is only available if the location services are enabled in addition.

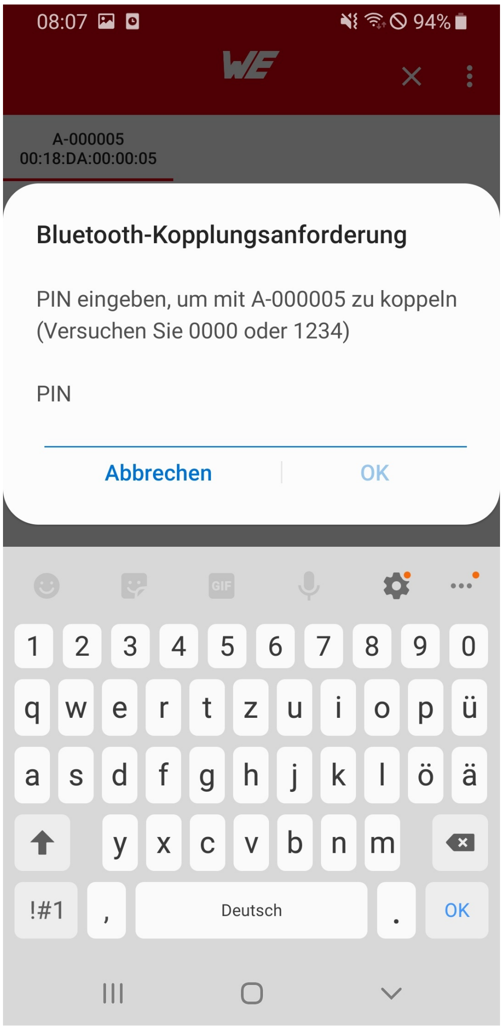
Android	iOS
• Press "Scan" to find the module on the radio.  08:07 94% Select device Found devices Scan Default ☆ A-000005 -63 dBm 00:18:DA:00:00:05 Scan Terminal Info	 13:08 Found devices **Demo Device** 6D:69:6E:61:74:65 **A-000002** 0B:FB:43:99:84:80 -61dBm Scan Terminal Info

- When the module A-xxxxxx appears, press connect. (Note: the part after "A-" is the 3 LSB as ASCII hex of the BTMAC, the fixed part "0x0018DA" is not part of the device descriptor).
- As soon as the module has received the connection request the module *LED_1* (*LED_3* on the Proteus-EV) will constantly light up.

<table>
<tr><th>Android</th><th>iOS</th></tr>
<tr><td colspan="2">

- Then the radio module requests for the static passkey. In default, the passkey is "123123".
- The Bluetooth® coupling requirement popup is shown in your smart phone.
- When the bonding feature is enabled in the authentication settings and the bonding information already exists, a re-entering of the passkey is not required when reconnecting.

</td></tr>
<tr><td>



08:07 94%

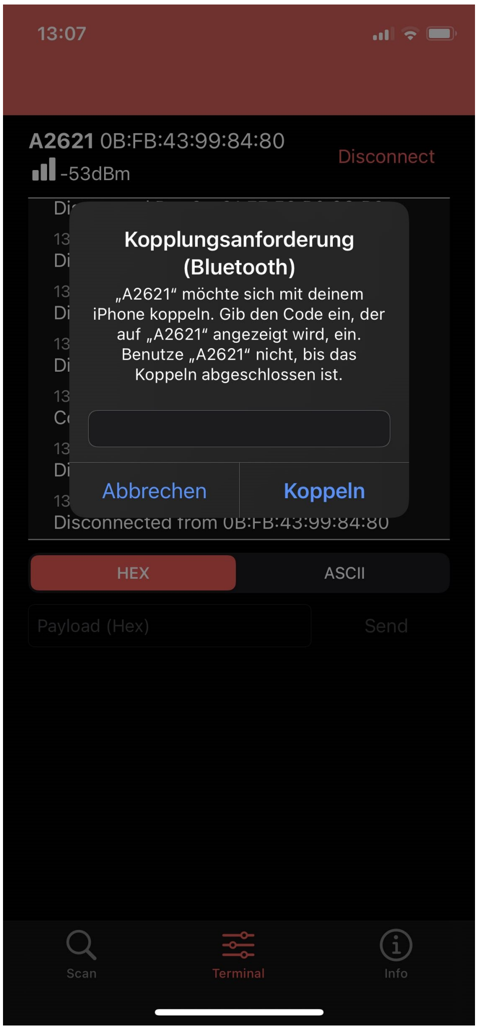
A-000005
00:18:DA:00:00:05

Bluetooth-Kopplungsanforderung

PIN eingeben, um mit A-000005 zu koppeln (Versuchen Sie 0000 oder 1234)

PIN

</td><td>



13:07

A2621 0B:FB:43:99:84:80
-53dBm

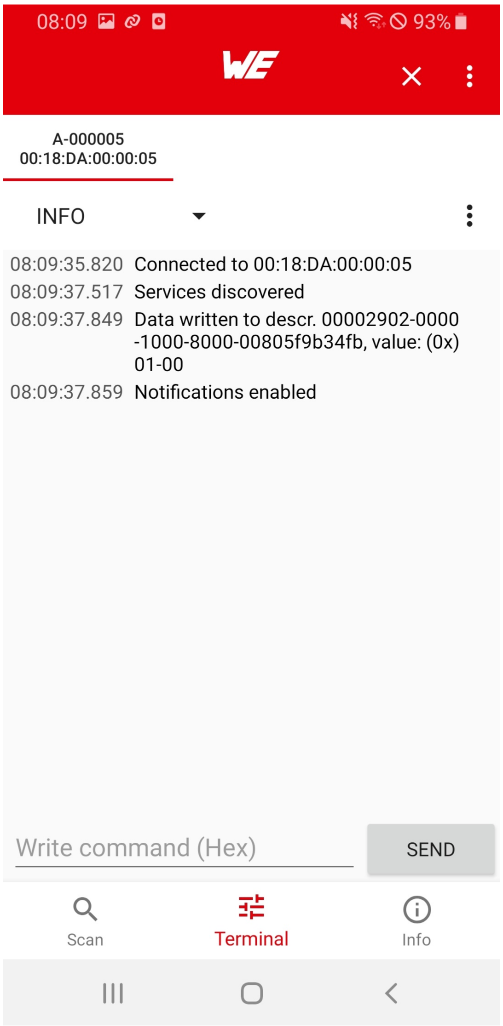
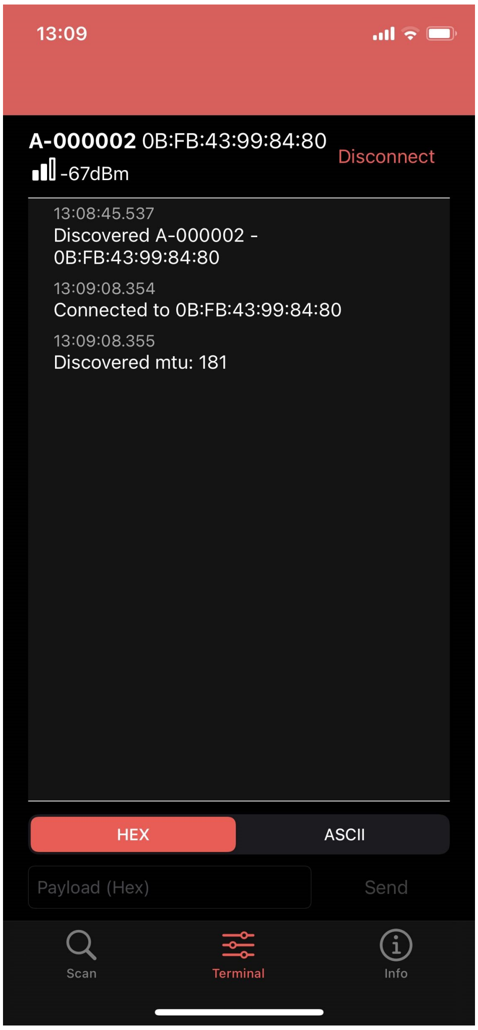
Kopplungsanforderung (Bluetooth)

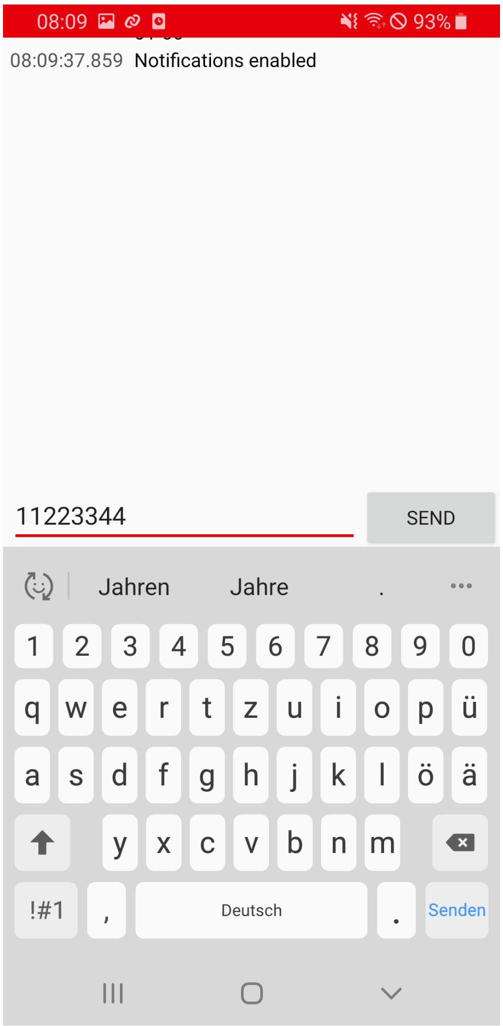
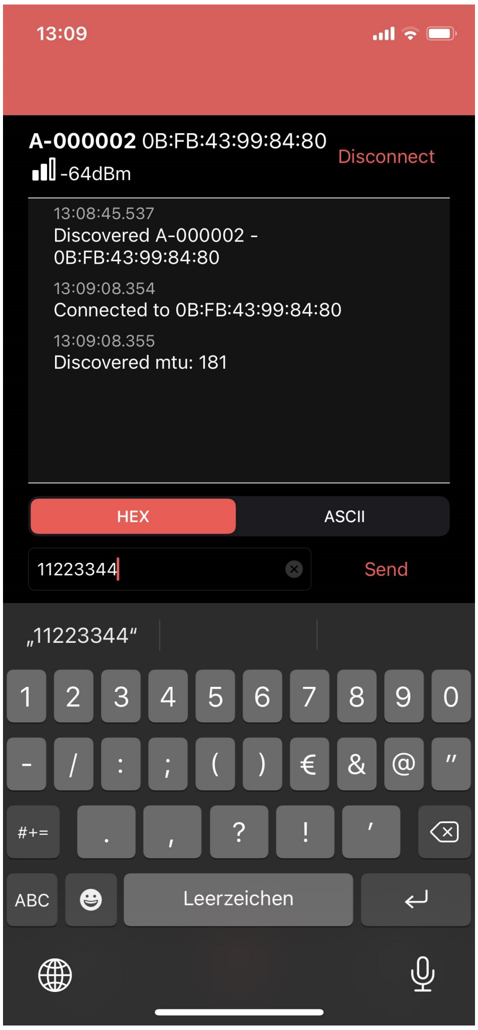
„A2621“ möchte sich mit deinem iPhone koppeln. Gib den Code ein, der auf „A2621“ angezeigt wird, ein. Benutze „A2621“ nicht, bis das Koppeln abgeschlossen ist.

Disconnected from 0B:FB:43:99:84:80

</td></tr>
</table>

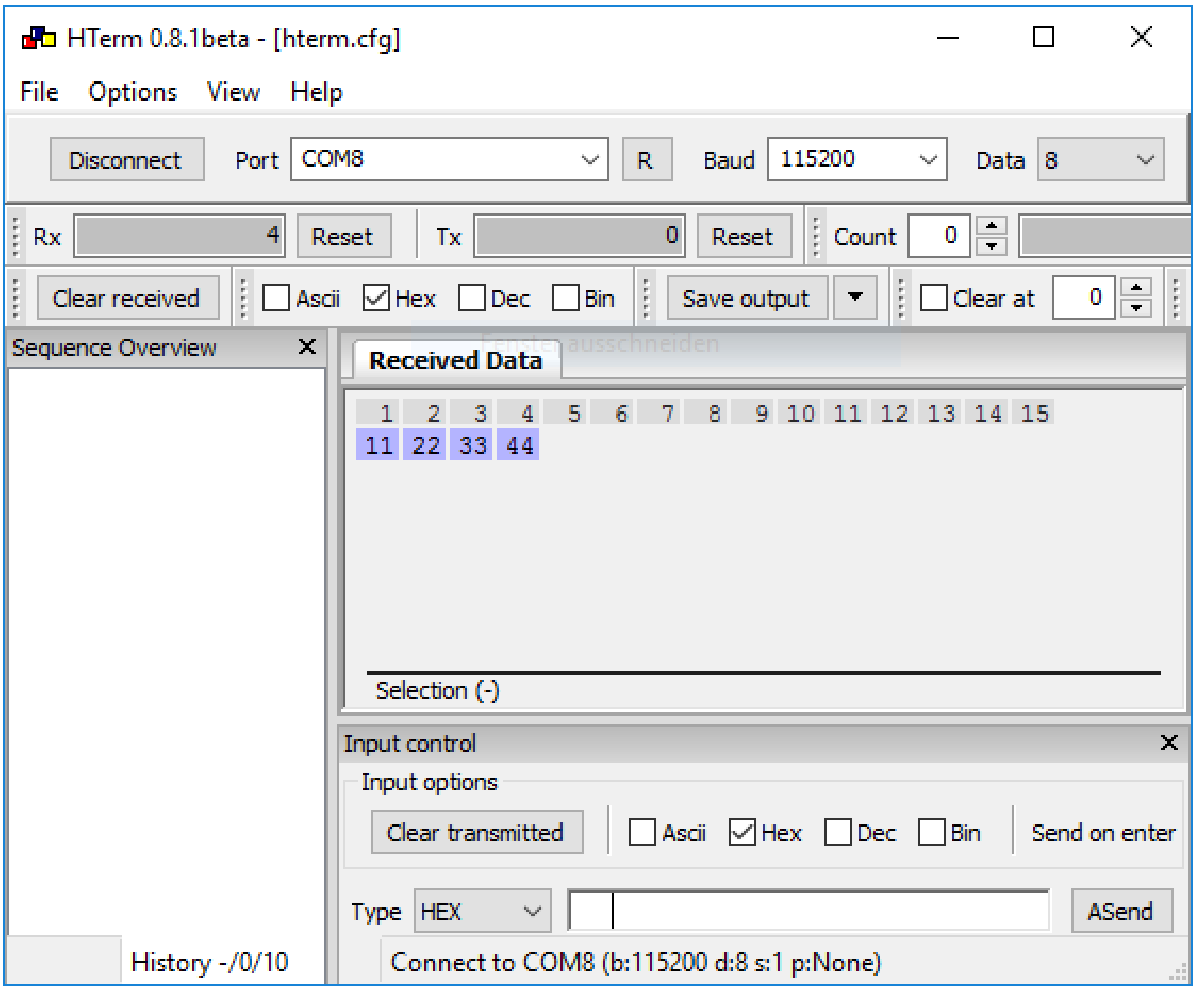
❗ In few cases the Android may show an "authentication timeout" pop-up message, when entering the key. In this case, please proceed entering the key and simply do a reconnect. On this reconnect, the entered key information is reused and the connection is opened.

Android	iOS
• Now you are authenticated and the *LED_2* (*LED_2* on the Proteus-EV) is turned on. Now data can be transmitted in both directions.	
 08:09 93% A-000005 00:18:DA:00:00:05 INFO Write command (Hex) SEND Scan Terminal Info 08:09:35.820 Connected to 00:18:DA:00:00:05 08:09:37.517 Services discovered 08:09:37.849 Data written to descr. 00002902-0000-1000-8000-00805f9b34fb, value: (0x) 01-00 08:09:37.859 Notifications enabled	 13:09 A-000002 0B:FB:43:99:84:80 -67dBm Disconnect HEX ASCII Payload (Hex) Send Scan Terminal Info 13:08:45.537 Discovered A-000002 - 0B:FB:43:99:84:80 13:09:08.354 Connected to 0B:FB:43:99:84:80 13:09:08.355 Discovered mtu: 181

Android	iOS
• First of all, we want to send data from the smart phone to the radio module. To do so, enter your payload (for example 0x11 0x22 0x33 0x44) and press "SEND". The allowed payload size is dependent on the MTU that was negotiated in the connection process. The smallest supported MTU for all Bluetooth® 4.0 (or newer) devices results in a max payload of 19 bytes.	
• Android usually allows up to 243 bytes.  08:09:37.859 Notifications enabled 11223344 	• iOS usually allows up to 181 bytes  A-000002 0B:FB:43:99:84:80 -64dBm 13:08:45.537 Discovered A-000002 - 0B:FB:43:99:84:80 13:09:08.354 Connected to 0B:FB:43:99:84:80 13:09:08.355 Discovered mtu: 181 11223344

Android	iOS

- The payload that has been sent via radio is output by the Proteus via UART. In peripheral only mode, a transparent UART interface is used. This means, that only payload data is transmitted, without any packet header or footer. Thus the transmitted bytes 0x11 0x22 0x33 0x44 are displayed on the connected terminal program.



Port COM8 Baud 115200 Data 8

Rx 4 Tx 0 Count 0 Clear at 0

```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
11 22 33 44
```

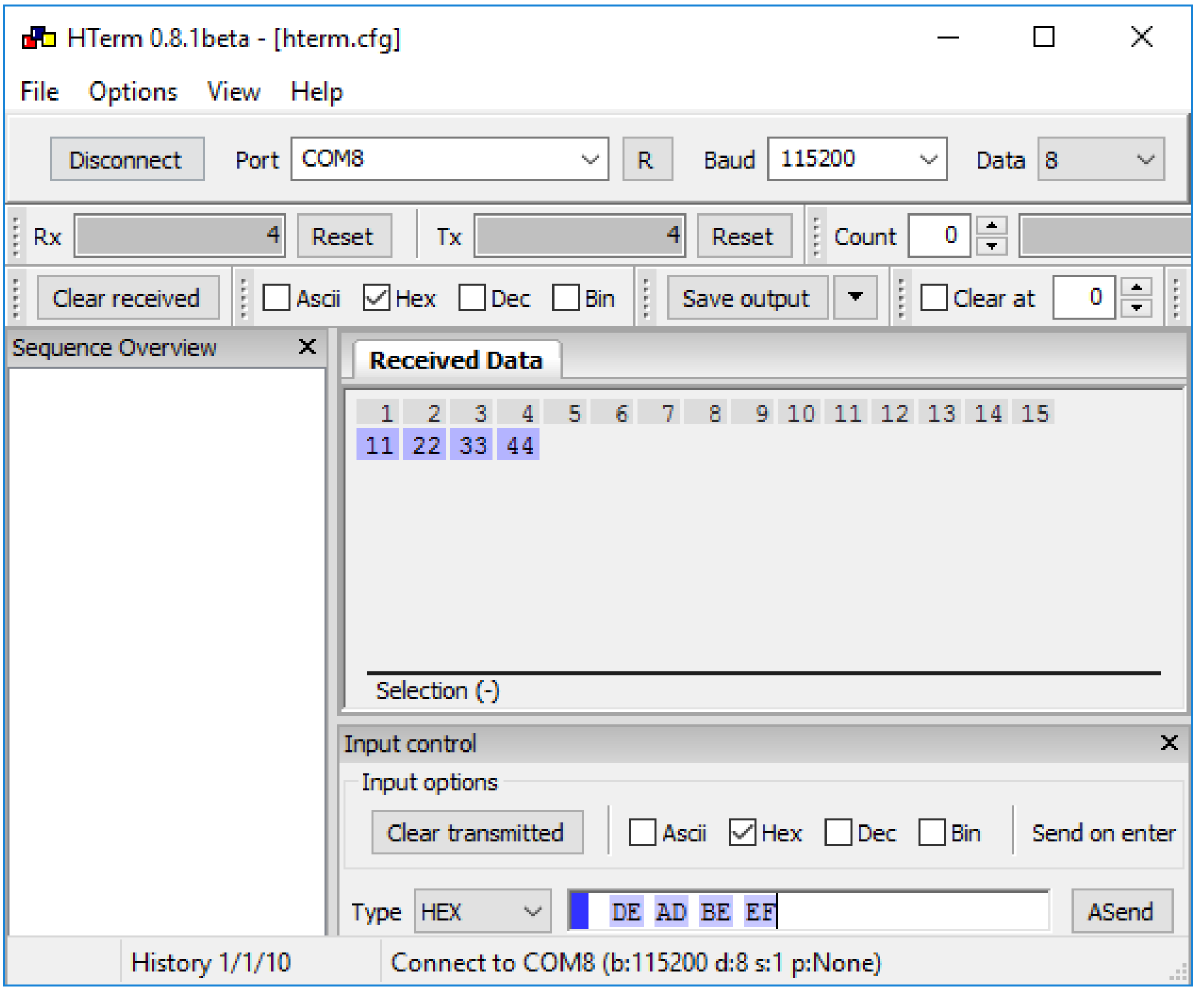
Selection (-)

Type HEX

Connect to COM8 (b:115200 d:8 s:1 p:None)

Android	iOS

- To send back data simply enter your payload in the respective terminal program field and press enter. In this example we choose 0xDE 0xAD 0xBE 0xEF. The header 0x01 will be automatically applied by the module and is not to be transmitted by the host.
- Here again the maximum payload size (MTU) must be respected.



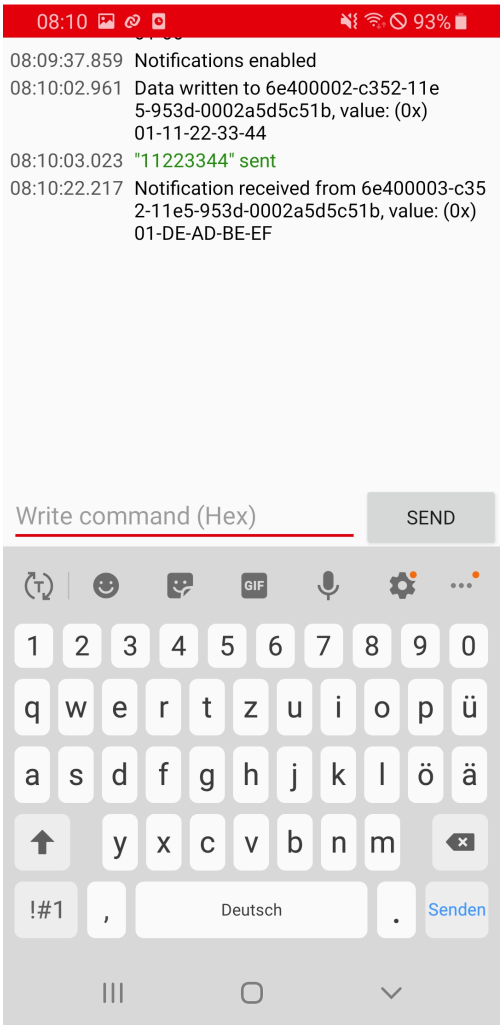
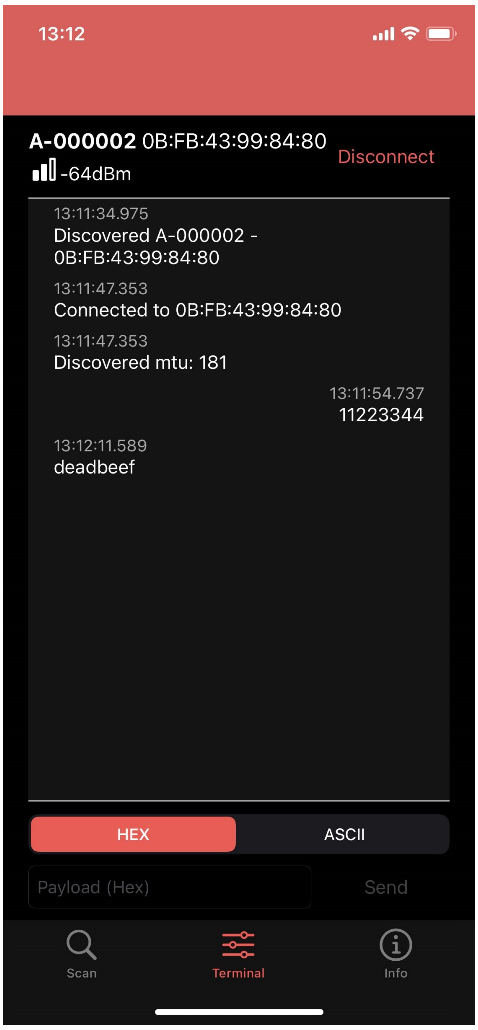
Port COM8
Baud 115200
Data 8
Rx 4
Tx 4
Count 0
□ Ascii ☑ Hex □ Dec □ Bin
□ Clear at 0

```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
11 22 33 44
```

□ Ascii ☑ Hex □ Dec □ Bin

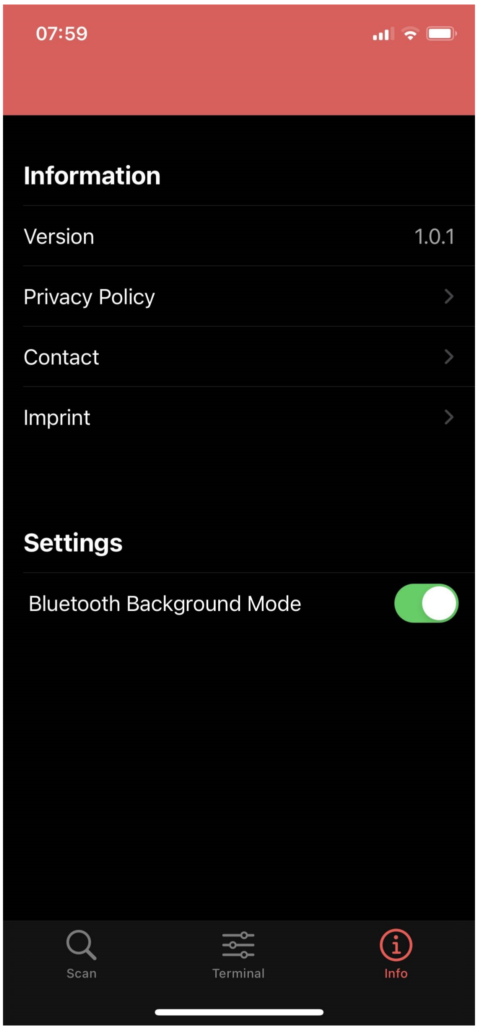
Type HEX   DE AD BE EF

History 1/1/10   Connect to COM8 (b:115200 d:8 s:1 p:None)

Android	iOS
• The received data is shown in the status window. It contains the header byte 0x01 and the payload 0xDE 0xAD 0xBE 0xEF, that has been entered in the terminal program.	• The received data is shown in the status window.
 08:09:37.859 Notifications enabled 08:10:02.961 Data written to 6e400002-c352-11e5-953d-0002a5d5c51b, value: (0x) 01-11-22-33-44 08:10:03.023 "11223344" sent 08:10:22.217 Notification received from 6e400003-c352-11e5-953d-0002a5d5c51b, value: (0x) 01-DE-AD-BE-EF 08:10 93% Write command (Hex) SEND Deutsch Senden	 A-000002 0B:FB:43:99:84:80 -64dBm 13:11:34.975 Discovered A-000002 - 0B:FB:43:99:84:80 13:11:47.353 Connected to 0B:FB:43:99:84:80 13:11:47.353 Discovered mtu: 181 13:11:54.737 11223344 13:12:11.589 deadbeef 13:12 Disconnect HEX ASCII Payload (Hex) Send Scan Terminal Info

4.2.1 Background service on iOS

By default, iOS disconnects the Bluetooth® LE connection, in case the WE Bluetooth LE Terminal App is put to background. To avoid this behavior, the background service of the WE Bluetooth LE Terminal App must be enabled by going to the info tab and selecting the "Bluetooth Background Mode" slider.



07:59

Information

Version 1.0.1

Privacy Policy

Contact

Imprint

Settings

Bluetooth Background Mode

Figure 6: Enable the background service on iOS

4.3 Proteus module or USB radio stick as central device

This chapter describes how to setup a connection to the Proteus radio module in peripheral mode (factory state), when another Proteus radio module or even Proteus USB radio stick is used as central device.

! For reasons of simplicity, we will call the Proteus radio module or USB radio stick, that is intended to setup the connection to the Proteus module running in peripheral only mode, **Proteus_central**. Furthermore, we will call the Proteus module running in peripheral only mode, **Proteus_peripheral**.

! Please note that the **Proteus_central** must run in command mode to initiate the connection setup.

! In this example we assume that the MAC of the **Proteus_peripheral** is 0x0018DA000011.

1. Configuring the correct security mode of the **Proteus_central**:
   The **Proteus_peripheral** uses the "static passkey pairing with bonding" as default security mode. As the central device must use the same security mode, the user setting `RF_SecFlags` of the **Proteus_central** must be also set to "static passkey with bonding" (0x0B = 11), before a connection setup can be done. To do so, please send the following command (`CMD_SET_REQ` with settings index 0x0C and value 0x0B) to the **Proteus_central**:

Info	**Proteus_central**	**Proteus_peripheral**
⇒ Request `CMD_SET_REQ` to set the right security mode of the **Proteus_central**	02 11 02 00 0C 0B 16	
⇐ Response `CMD_SET_CNF`: Setting successfully set	02 51 01 00 00 52	
⇐ Response `CMD_GETSTATE_CNF`: **Proteus_central** restarted	02 41 02 00 01 01 41	

   Now, the connection setup can be initiated.

2. Connect **Proteus_central** to the **Proteus_peripheral** via Bluetooth® LE.

Info	**Proteus_central**	**Proteus_peripheral**
⇒ Request CMD_CONNECT_REQ with FS_BTMAC of **Proteus_peripheral**	02 06 06 00 11 00 00 DA 18 00 D1	
⇐ Response CMD_CONNECT_CNF: Request understood, try to connect now	02 46 01 00 00 45	
⇐ Indication CMD_CONNECT_IND: Physical connection established successfully to the module with FS_BTMAC 0x11 0x00 0x00 0xDA 0x18 0x00	02 86 07 00 00 11 00 00 DA 18 00 50	

a) Option A: No bonding data available (i.e. when connecting for the first time). Pass key must be entered as soon as requested by the **Proteus_central** by a CMD_PASSKEY_IND message.

In case the CMD_PASSKEY_IND message does not appear, but the Bluetooth® LE connection has been closed, the security settings of the **Proteus_central** do not match. Please check again the user setting RF_SecFlags of the **Proteus_central**, as described in step 1.

Info	**Proteus_central**	**Proteus_peripheral**
⇐ Indication CMD_PASSKEY_IND to ask for the pass key	02 8D 07 00 00 11 00 00 DA 18 00 5B	
⇒ Answer with the CMD_PASSKEY_REQ and the correct pass key (default is "123123")	02 0D 06 00 31 32 33 31 32 33 09	
⇐ Response CMD_PASSKEY_CNF: Pass key ok	02 4D 01 00 00 4E	
⇐ Indication CMD_SECURITY_IND, status 0x01 (encrypted link, bonding established), with FS_BTMAC 0x11 0x00 0x00 0xDA 0x18 0x00	02 88 07 00 01 11 00 00 DA 18 00 5F	
⇐ Indication CMD_CHANNELOPEN_RSP: Channel opened successfully to the module with FS_BTMAC 0x11 0x00 0x00 0xDA 0x18 0x00 and maximum payload size of 0xF3 (243 Bytes) per packet	02 C6 08 00 00 11 00 00 DA 18 00 F3 EC	

b) Option B: Bonding data is already available (i.e. when reconnecting). No pass key must be entered.

Info	**Proteus_central**	**Proteus_peripheral**
⇐ Indication `CMD_SECURITY_IND`, status 0x00 (encrypted link, bonding data already available), with `FS_BTMAC` 0x11 0x00 0x00 0xDA 0x18 0x00	02 88 07 00 00 11 00 00 DA 18 00 5E	
⇐ Indication `CMD_CHANNELOPEN_RSP`: Channel opened successfully to the module with `FS_BTMAC` 0x11 0x00 0x00 0xDA 0x18 0x00 and maximum payload size of 0xF3 (243 Bytes) per packet	02 C6 08 00 00 11 00 00 DA 18 00 F3 EC	

3. Now the connection is active. Thus data can be sent in each direction. Let us send a string "ABCD" from **Proteus_peripheral** to **Proteus_central**.

The RSSI values will be different in your tests.

Info	**Proteus_central**	**Proteus_peripheral**
⇒ Transparent send "ABCD" to **Proteus_central**		41 42 43 44
⇐ Indication `CMD_DATA_IND`: Received string "ABCD" from `FS_BTMAC` 0x11 0x00 0x00 0xDA 0x18 0x00 with RSSI of 0xCA (-54dBm)	02 84 0B 00 11 00 00 DA 18 00 CA 41 42 43 44 90	

4. Reply with "EFGH" to the **Proteus_peripheral**.

Info	**Proteus_central**	**Proteus_peripheral**
⇒ Request `CMD_DATA_REQ`: Send "EFGH" to **Proteus_peripheral**	02 04 04 00 45 46 47 48 0E	
⇐ Response `CMD_DATA_CNF`: Request received, send data now	02 44 01 00 00 47	
⇐ Transparent received string "EFGH"		45 46 47 48
⇐ Response `CMD_TXCOMPLETE_RSP`: Data transmitted successfully	02 C4 01 00 00 C7	

5. Now **Proteus_central** closes the connection.

Info	**Proteus_central**	**Proteus_peripheral**
⇒ Request `CMD_DISCONNECT_REQ`: Disconnect	02 07 00 00 05	
⇐ Response `CMD_DISCONNECT_CNF`: Request received, disconnect now	02 47 01 00 00 44	
⇐ Indication `CMD_DISCONNECT_IND`: Connection closed	02 87 01 00 16 92	

5 References

[1] Würth Elektronik. Application note 2 - Proteus-I advanced developer guide. http://www.we-online.com/ANR002.

[2] Würth Elektronik. Application note 5 - Proteus-II advanced developer guide. http://www.we-online.com/ANR005.

[3] Würth Elektronik. Application note 9 - Proteus-III(-SPI) advanced developer guide. http://www.we-online.com/ANR009.

[4] Würth Elektronik. WE Bluetooth LE Terminal app for Android. https://play.google.com/store/apps/details?id=com.eisos.android.terminal.

[5] Würth Elektronik. WE Bluetooth LE Terminal app for iOS. https://apps.apple.com/de/app/proteus-connect/id1533941485.

[6] Würth Elektronik. Source code of WE Bluetooth LE Terminal app (cross platform). https://github.com/WurthElektronik/Proteus-Connect.

6 Important notes

The Application Note and its containing information ("Information") is based on Würth Elektronik eiSos GmbH & Co. KG and its subsidiaries and affiliates ("WE eiSos") knowledge and experience of typical requirements concerning these areas. It serves as general guidance and shall not be construed as a commitment for the suitability for customer applications by WE eiSos. While WE eiSos has used reasonable efforts to ensure the accuracy of the Information, WE eiSos does not guarantee that the Information is error-free, nor makes any other representation, warranty or guarantee that the Information is completely accurate or up-to-date. The Information is subject to change without notice. To the extent permitted by law, the Information shall not be reproduced or copied without WE eiSos' prior written permission. In any case, the Information, in full or in parts, may not be altered, falsified or distorted nor be used for any unauthorized purpose.

WE eiSos is not liable for application assistance of any kind. Customer may use WE eiSos' assistance and product recommendations for customer's applications and design. No oral or written Information given by WE eiSos or its distributors, agents or employees will operate to create any warranty or guarantee or vary any official documentation of the product e.g. data sheets and user manuals towards customer and customer shall not rely on any provided Information. THE INFORMATION IS PROVIDED "AS IS". CUSTOMER ACKNOWLEDGES THAT WE EISOS MAKES NO REPRESENTATIONS AND WARRANTIES OF ANY KIND RELATED TO, BUT NOT LIMITED TO THE NON-INFRINGEMENT OF THIRD PARTIES' INTELLECTUAL PROPERTY RIGHTS OR THE MERCHANTABILITY OR FITNESS FOR A PURPOSE OR USAGE. WE EISOS DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT RELATING TO ANY COMBINATION, MACHINE, OR PROCESS IN WHICH WE EISOS INFORMATION IS USED. INFORMATION PUBLISHED BY WE EISOS REGARDING THIRD-PARTY PRODUCTS OR SERVICES DOES NOT CONSTITUTE A LICENSE FROM WE eiSos TO USE SUCH PRODUCTS OR SERVICES OR A WARRANTY OR ENDORSEMENT THEREOF.

The responsibility for the applicability and use of WE eiSos' components in a particular customer design is always solely within the authority of the customer. Due to this fact it is up to the customer to evaluate and investigate, where appropriate, and decide whether the device with the specific characteristics described in the specification is valid and suitable for the respective customer application or not. The technical specifications are stated in the current data sheet and user manual of the component. Therefore the customers shall use the data sheets and user manuals and are cautioned to verify that they are current. The data sheets and user manuals can be downloaded at *www.we-online.com*. Customers shall strictly observe any product-specific notes, cautions and warnings. WE eiSos reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time without notice.

WE eiSos will in no case be liable for customer's use, or the results of the use, of the components or any accompanying written materials. IT IS CUSTOMER'S RESPONSIBILITY TO VERIFY THE RESULTS OF THE USE OF THIS INFORMATION IN IT'S OWN PARTICULAR ENGINEERING AND PRODUCT ENVIRONMENT AND CUSTOMER ASSUMES THE ENTIRE RISK OF DOING SO OR FAILING TO DO SO. IN NO CASE WILL WE EISOS BE LIABLE FOR CUSTOMER'S USE, OR THE RESULTS OF IT'S USE OF THE COMPONENTS OR ANY ACCOMPANYING WRITTEN MATERIAL IF CUSTOMER TRANSLATES, ALTERS, ARRANGES, TRANSFORMS, OR OTHERWISE MODIFIES THE INFORMATION IN ANY WAY, SHAPE OR FORM.

If customer determines that the components are valid and suitable for a particular design and wants to order the corresponding components, customer acknowledges to minimize the risk of loss and harm to individuals and bears the risk for failure leading to personal injury or death due to customers usage of the components. The components have been designed and developed for usage in general electronic equipment only. The components are not authorized for use in equipment where a higher safety standard and reliability standard is especially required or where a failure of the components is reasonably expected to cause severe personal injury or death, unless WE eiSos and customer have executed an agreement specifically governing such use. Moreover WE eiSos components are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation, transportation signal, disaster prevention, medical, public information network etc. WE eiSos must be informed about the intent of such usage before the design-in stage. In addition, sufficient reliability evaluation checks for safety must be performed on every component which is used in electrical circuits that require high safety and reliability functions or performance. COSTUMER SHALL INDEMNIFY WE EISOS AGAINST ANY DAMAGES ARISING OUT OF THE USE OF THE COMPONENTS IN SUCH SAFETY-CRITICAL APPLICATIONS.

List of Figures

List of Tables

Contact
Würth Elektronik eiSos GmbH & Co. KG
Division Wireless Connectivity & Sensors

Max-Eyth-Straße 1
74638 Waldenburg
Germany

Tel.: +49 651 99355-0
Fax.: +49 651 99355-69
www.we-online.com/wireless-connectivity

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT