



USER MANUAL

PROTEUS-E

2612011024000

VERSION 1.8

JANUARY 13, 2026

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT

MUST READ

Check for firmware updates

Before using the product, make sure you use the most recent firmware version, data sheet, and user manual. This is especially important for Wireless Connectivity products that were not purchased directly from Würth Elektronik eiSos. A firmware update on these respective products may be required.

We strongly recommend including the possibility of a firmware update in the customer system design.

Contents

| | |
|---|-----------|
| Overview of helpful application notes | 8 |
| 1. Revision history | 10 |
| 2. Abbreviations | 11 |
| 3. Introduction | 14 |
| 3.1. Operational description | 14 |
| 3.1.1. Key features | 15 |
| 3.1.2. Connectivity | 16 |
| 3.2. Block diagram | 17 |
| 3.3. Ordering information | 17 |
| 4. Electrical specifications | 18 |
| 4.1. Operating conditions | 18 |
| 4.2. Absolute maximum ratings | 18 |
| 4.3. Power consumption | 19 |
| 4.3.1. Static | 19 |
| 4.3.2. Dynamic | 21 |
| 4.4. Radio characteristics | 23 |
| 4.5. Pin characteristics | 24 |
| 5. Pinout | 25 |
| 6. Quick start | 27 |
| 6.1. Minimal pin connections | 27 |
| 6.2. Antenna connection | 29 |
| 6.2.1. On-board PCB antenna | 29 |
| 6.2.2. External antenna | 29 |
| 6.3. Power up | 30 |
| 6.4. Quickstart example | 31 |
| 7. Functional description | 33 |
| 7.1. Operation modes | 33 |
| 7.2. Radio module states | 33 |
| 7.3. State indication using the LED pins | 35 |
| 7.4. Sleep mode | 35 |
| 7.5. Identification of a Proteus-e device on the radio | 36 |
| 7.6. Connection based data transmission | 36 |
| 7.7. Advertising of custom data | 39 |
| 7.7.1. Restrictions | 39 |
| 7.7.2. Application of custom advertising and scan response data | 39 |
| 7.8. Energy-efficient distance estimation solutions | 40 |
| 7.9. Configure the module for low power consumption | 40 |
| 7.10. Start the direct test mode (DTM) | 41 |
| 7.11. Using the 2 MBit phy | 43 |

| | |
|---|-----------|
| 8. Host connection | 45 |
| 8.1. Serial interface: UART | 45 |
| 8.1.1. Reset behavior | 45 |
| 9. The command interface | 46 |
| 9.1. Setup connections | 48 |
| 9.1.1. CMD_CONNECT_IND | 48 |
| 9.1.2. CMD_SECURITY_IND | 48 |
| 9.1.3. CMD_CHANNELOPEN_RSP | 48 |
| 9.1.4. CMD_DISCONNECT_REQ | 49 |
| 9.1.5. CMD_DISCONNECT_IND | 49 |
| 9.1.6. CMD_PHYUPDATE_REQ | 50 |
| 9.1.7. CMD_PHYUPDATE_IND | 50 |
| 9.1.8. CMD_GETBONDS_REQ | 51 |
| 9.1.8.1. Example 1 | 51 |
| 9.1.9. CMD_DELETEBONDS_REQ | 52 |
| 9.1.9.1. Example 1 | 52 |
| 9.1.9.2. Example 2 | 53 |
| 9.1.10. CMD_ALLOWUNBONDEDCONNECTIONS_REQ | 53 |
| 9.2. Transmit and receive data | 54 |
| 9.2.1. CMD_DATA_REQ | 54 |
| 9.2.2. CMD_TXCOMPLETE_RSP | 54 |
| 9.2.3. CMD_DATA_IND | 54 |
| 9.3. Configuring the module and modifying the device settings | 56 |
| 9.3.1. CMD_SET_REQ | 56 |
| 9.3.1.1. Example 1 | 57 |
| 9.3.1.2. Example 2 | 57 |
| 9.3.2. CMD_GET_REQ | 58 |
| 9.3.2.1. Example 1 | 58 |
| 9.3.3. CMD_SETRAM_REQ | 59 |
| 9.3.3.1. Example 1 | 59 |
| 9.3.4. CMD_GETRAM_REQ | 60 |
| 9.3.4.1. Example 1 | 60 |
| 9.4. Manage the device state | 61 |
| 9.4.1. CMD_GETSTATE_REQ | 61 |
| 9.4.1.1. Example 1 | 61 |
| 9.4.2. CMD_RESET_REQ | 62 |
| 9.4.3. CMD_SLEEP_REQ | 62 |
| 9.4.4. CMD_SLEEP_IND | 63 |
| 9.4.5. CMD_FACTORYRESET_REQ | 63 |
| 9.4.6. CMD_UARTDISABLE_REQ | 64 |
| 9.4.7. CMD_UARTENABLE_IND | 65 |
| 9.5. Run the Bluetooth test modes | 66 |
| 9.5.1. CMD_DTMSTART_REQ | 66 |
| 9.5.2. CMD_DTM_REQ | 66 |
| 9.5.2.1. Example: Transmission, 16 times 0x0F, channel 0 | 68 |
| 9.5.2.2. Example: Receiver, channel 0 | 69 |
| 9.5.2.3. Example: Transmission, carrier test, channel 0 | 69 |

| | | |
|------------|--|-----------|
| 9.5.2.4. | Example: Set TX power to -4 dBm | 70 |
| 9.5.2.5. | Example: Set PHY to 2MBit mode | 70 |
| 9.6. | Switching GPIOs by remote control | 71 |
| 9.6.1. | CMD_GPIO_LOCAL_WRITECONFIG_REQ | 71 |
| 9.6.1.1. | Example: Configure two GPIOs to output high | 72 |
| 9.6.2. | CMD_GPIO_LOCAL_READCONFIG_REQ | 74 |
| 9.6.2.1. | Example: Read the current GPIO configuration | 75 |
| 9.6.3. | CMD_GPIO_REMOTE_WRITECONFIG_REQ | 76 |
| 9.6.3.1. | Example: Configure two GPIOs of the connected remote device to output high | 77 |
| 9.6.4. | CMD_GPIO_REMOTE_READCONFIG_REQ | 78 |
| 9.6.4.1. | Example: Read the current GPIO configuration of the connected remote device | 79 |
| 9.6.5. | CMD_GPIO_REMOTE_WRITE_REQ | 80 |
| 9.6.5.1. | Example: Set a remote output GPIO to low | 81 |
| 9.6.6. | CMD_GPIO_REMOTE_READ_REQ | 82 |
| 9.6.6.1. | Example: Read the values of remote GPIOs | 83 |
| 9.6.7. | CMD_GPIO_LOCAL_WRITE_REQ | 84 |
| 9.6.7.1. | Example: Set a local output GPIO to low | 85 |
| 9.6.8. | CMD_GPIO_LOCAL_READ_REQ | 86 |
| 9.6.8.1. | Example: Read the values of local GPIOs | 87 |
| 9.6.9. | CMD_GPIO_REMOTE_WRITECONFIG_IND | 88 |
| 9.6.9.1. | Example: Two GPIOs have been configured by the connected re- mote device to output high | 88 |
| 9.6.10. | CMD_GPIO_REMOTE_WRITE_IND | 89 |
| 9.6.10.1. | Example: GPIOs have been written via remote access | 89 |
| 9.6.11. | CMD_GPIO_LOCAL_WRITE_IND | 90 |
| 9.6.11.1. | Example: GPIOs of the remote device have been written by its local host | 90 |
| 9.7. | Other messages | 91 |
| 9.7.1. | CMD_ERROR_IND | 91 |
| 9.8. | Message overview | 92 |
| 10. | User settings - Module configuration values | 95 |
| 10.1. | FS_DeviceInfo: Read the chip type and OS version | 95 |
| 10.1.1. | Example 1 | 96 |
| 10.2. | FS_FWVersion: Read the firmware version | 97 |
| 10.2.1. | Example 1 | 97 |
| 10.3. | FS_MAC: Read the MAC address | 98 |
| 10.3.1. | Example 1 | 98 |
| 10.4. | FS_BTMAC: Modify the Bluetooth conform MAC address | 99 |
| 10.4.1. | Example 1 | 99 |
| 10.4.2. | Example 2 | 99 |
| 10.5. | FS_SerialNumber: Read the serial number of the module | 101 |
| 10.5.1. | Example 1 | 101 |
| 10.6. | RF_DeviceName: Modify the device name | 102 |
| 10.6.1. | Example 1 | 102 |
| 10.6.2. | Example 2 | 102 |

| | |
|--|-----|
| 10.7. RF_StaticPasskey: Modify the static passkey | 104 |
| 10.7.1. Example 1 | 104 |
| 10.7.2. Example 2 | 104 |
| 10.8. RF_SecFlags: Modify the security settings | 105 |
| 10.8.1. Example 1 | 106 |
| 10.8.2. Example 2 | 106 |
| 10.9. RF_ScanResponseData: Modify the content of the scan response packet | 108 |
| 10.9.1. Example 1 | 108 |
| 10.9.2. Example 2 | 109 |
| 10.10. RF_AdvertisingData: Modify the content of the advertising packet | 110 |
| 10.10.1. Example 1 | 110 |
| 10.10.2. Example 2 | 111 |
| 10.11. RF_AdvertisingTimeout: Modify the advertising timeout | 112 |
| 10.11.1. Example 1 | 112 |
| 10.11.2. Example 2 | 112 |
| 10.12. RF_AdvertisingInterval: Modify the advertising interval | 113 |
| 10.12.1. Example 1 | 113 |
| 10.12.2. Example 2 | 113 |
| 10.13. RF_ConnectionInterval: Modify the connection interval | 114 |
| 10.13.1. Example 1 | 115 |
| 10.13.2. Example 2 | 115 |
| 10.14. RF_TXPower: Modify the output power | 116 |
| 10.14.1. Example 1 | 116 |
| 10.14.2. Example 2 | 116 |
| 10.15. RF_SPPBaseUUID: Configure the SPP base UUID | 118 |
| 10.15.1. Example 1 | 118 |
| 10.15.2. Example 2 | 118 |
| 10.16. RF_SPPServiceUUID: Configure the SPP service UUID | 120 |
| 10.16.1. Example 1 | 120 |
| 10.16.2. Example 2 | 120 |
| 10.17. RF_SPPRXUUID: Configure the SPP RX UUID | 121 |
| 10.17.1. Example 1 | 121 |
| 10.17.2. Example 2 | 121 |
| 10.18. RF_SPPTXUUID: Configure the SPP TX UUID | 122 |
| 10.18.1. Example 1 | 122 |
| 10.18.2. Example 2 | 122 |
| 10.19. RF_Appearance: Configure the appearance of the device | 123 |
| 10.19.1. Example 1 | 123 |
| 10.19.2. Example 2 | 123 |
| 10.20. UART_ConfigIndex: Modify the UART speed | 124 |
| 10.20.1. Example 1 | 126 |
| 10.20.2. Example 2 | 126 |
| 10.21. UART_TransparentTimeout: Modify the transparent mode timeout | 127 |
| 10.21.1. Example 1 | 127 |
| 10.21.2. Example 2 | 127 |
| 10.22. UART_TransparentMaxPayload: Modify the transparent mode max payload | 128 |
| 10.22.1. Example 1 | 128 |
| 10.22.2. Example 2 | 128 |

| | |
|--|------------|
| 10.23. UART_TransparentETX: Modify the transparent ETX characters | 129 |
| 10.23.1. Example 1 | 129 |
| 10.23.2. Example 2 | 129 |
| 10.24. UART_TransparentETXConfig: Configure the usage of the ETX in transparent mode | 130 |
| 10.24.1. Example 1 | 130 |
| 10.24.2. Example 2 | 130 |
| 10.25. CFG_Flags: Configure the module | 132 |
| 10.25.1. Example 1 | 132 |
| 10.25.2. Example 2 | 132 |
| 11. Timing parameters | 136 |
| 11.1. Reset and sleep | 136 |
| 11.2. Bluetooth LE timing parameters | 136 |
| 11.3. Connection establishment | 136 |
| 11.4. Connection based data transmission | 137 |
| 11.4.1. Maximum data throughput | 137 |
| 12. Transparent mode | 139 |
| 12.1. Reasons to use the transparent mode | 139 |
| 12.2. How to use the transparent mode | 139 |
| 12.3. More information | 139 |
| 12.3.1. UART | 139 |
| 13. Remote GPIO control | 142 |
| 13.1. Supported GPIO_IDs for remote and local control | 147 |
| 14. Customizing the Proteus-e | 148 |
| 14.1. UUID | 148 |
| 14.2. Appearance | 148 |
| 15. Custom firmware and configuration | 149 |
| 15.1. Custom configuration of standard firmware | 149 |
| 15.2. Customer specific firmware | 149 |
| 15.3. Customer firmware | 149 |
| 16. Firmware updates | 151 |
| 16.1. Firmware flashing using the production interface | 151 |
| 17. Firmware history | 152 |
| 17.1. Known issues | 152 |
| 18. Hardware history | 153 |
| 19. Design in guide | 154 |
| 19.1. Advice for schematic and layout | 154 |
| 19.2. Designing the antenna connection | 156 |
| 19.3. Antenna solutions | 157 |
| 19.3.1. Wire antenna | 157 |

| | | |
|------------|--|------------|
| 19.3.2. | Chip antenna | 157 |
| 19.3.3. | PCB antenna | 158 |
| 19.3.4. | Antennas provided by Würth Elektronik eiSos | 158 |
| 19.3.4.1. | 2600130021 - Himalia dipole antenna | 159 |
| 20. | Reference design | 160 |
| 20.1. | EV-Board | 161 |
| 20.1.1. | Schematic | 161 |
| 20.2. | Layout | 162 |
| 20.3. | Internal antenna radiation characteristics | 163 |
| 20.4. | Trace design | 164 |
| 20.4.1. | Simple short using internal antenna | 165 |
| 20.4.2. | 22 pF coupling capacitor using internal antenna | 166 |
| 20.4.3. | 22 pF coupling capacitor using external antenna | 168 |
| 20.5. | Antenna fine tuning | 169 |
| 21. | Manufacturing information | 170 |
| 21.1. | Moisture sensitivity level | 170 |
| 21.2. | Soldering | 170 |
| 21.2.1. | Reflow soldering | 170 |
| 21.2.2. | Cleaning | 171 |
| 21.2.3. | Potting and coating | 172 |
| 21.2.4. | Other notations | 172 |
| 21.3. | ESD handling | 172 |
| 21.4. | Safety recommendations | 173 |
| 22. | Product testing | 174 |
| 22.1. | Würth Elektronik eiSos in-house production tests | 174 |
| 22.2. | EMS production tests | 174 |
| 23. | Physical specifications | 176 |
| 23.1. | Dimensions | 176 |
| 23.2. | Weight | 176 |
| 23.3. | Light sensitivity | 176 |
| 23.4. | Module drawing | 177 |
| 23.5. | Footprint WE-FP-4+ | 178 |
| 23.6. | Antenna free area | 178 |
| 24. | Marking | 179 |
| 24.1. | Lot number | 179 |
| 24.2. | General labeling information | 180 |
| 25. | Information for explosion protection | 181 |
| 26. | References | 182 |
| 27. | Bluetooth SIG listing/qualification | 183 |

| | |
|---|------------|
| 28. Regulatory compliance information | 184 |
| 28.1. Important notice EU | 184 |
| 28.2. Important notice UKCA | 184 |
| 28.3. Important notice FCC | 184 |
| 28.4. Conformity assessment of the final product | 184 |
| 28.5. Exemption clause | 185 |
| 28.6. EU Declaration of conformity | 186 |
| 28.7. RED-DA Cybersecurity statement | 187 |
| 28.8. FCC Compliance Statement (US) | 188 |
| 28.8.1. FCC certificate | 188 |
| 28.9. IC Compliance Statement (Canada) | 189 |
| 28.9.1. IC certificate | 189 |
| 28.10. FCC and IC requirements to OEM integrators | 190 |
| 28.10.1. OEM requirements: | 190 |
| 28.10.2. Pre-certified antennas | 191 |
| 28.11. TELEC radio law approval (Japan) | 192 |
| 28.11.1. Label | 192 |
| 28.11.2. Certified antennas | 192 |
| 28.11.3. TELEC certificate | 193 |
| 28.12. ETA-WPC (India) | 194 |
| 28.12.1. ETA-WPC certificate | 194 |
| 28.13. NCC Compliance Statement (Taiwan) | 196 |
| 28.13.1. NCC certificate | 196 |
| 28.14. Certification of the end device | 198 |
| 29. Important notes | 199 |
| 30. Legal notice | 199 |
| 31. License terms | 200 |
| A. Additional CRC8 Information | 204 |
| A.1. Example CRC8 Implementation | 204 |
| A.2. CRC8 Test Vectors | 204 |
| B. Example code for host integration | 205 |

Overview of helpful application notes

Application note ANR008 - Wireless Connectivity Software Development Kit

<http://www.we-online.com/ANR008>

To ease the integration of the Würth Elektronik eiSos radio modules into an application, Würth Elektronik eiSos offers the corresponding Software Development Kit (SDK) for most commonly used host processors. This SDK contains drivers and examples in C-code to communicate with the corresponding radio module. This application note shows which SDKs are available and describes how to download and use them.

Application note ANR010 - Range estimation

<http://www.we-online.com/ANR010>

This application note presents the two most used mathematical range estimation models, Friis and two ray ground reflection, and its implementation in the range estimation tool of the RED-EXPERT.

Application note ANR024 - Proteus-e Advanced developer guide

<http://www.we-online.com/ANR024>

This advanced developer guide covers the details on the Proteus-e radio module that are required to implement compatible App for smart devices. It covers the documentation on the SPP-like Bluetooth® LE profile, the used protocols and data coding for arbitrary user payload. In addition all information required to develop a custom firmware on the Proteus module hardware platform are provided within.

Application note ANR025 - Proteus-e Quickstart

<http://www.we-online.com/ANR025>

The Proteus-e is a radio module based on the nRF52 Nordic Semiconductors SoC, which provides various Bluetooth LE and has its focus on low power features. This Application Note shows how easy the "transparent mode" can be used to quickly connect the Proteus-e in order to communicate with a Bluetooth LE connection partner (e.g. a smart device). As second step with more advanced functions, and therefore recommended to be used in applications, the reader is introduced to the Command Mode that is also available to be used on the Proteus-e.

Application note ANR027 - Bluetooth qualification guide

<http://www.we-online.com/ANR027>

Every product containing Bluetooth® technology needs to be qualified at the Bluetooth® SIG (special interest group). This application note explains the steps to be done to gain a Bluetooth® qualification for the end product using a Würth Elektronik eiSos Bluetooth® LE radio module.

Application note ANR030 - nRF Connect

<http://www.we-online.com/ANR030>

This application note gives a short overview about the options to create a custom firmware for Würth Elektronik eiSos radio modules by using the hardware platform and the embedded nRF5x system on chip. It presents options on firmware development environments and accessories (like SDKs) for the use within the nRF5 ecosystem. The reader is informed on how to access to a multitude of radio standards (like Bluetooth® LE, Bluetooth® MESH, Bluetooth® LE Audio, Matter, Zigbee, Thread, Wirepas) for custom firmware developments whilst the hardware platform can stay the same.

Ground plane effects on radio module antennas

<http://www.we-online.com/ANR033>

The ground plane plays a critical role in the performance of radio module antennas, affecting parameters such as radiation pattern, gain, and efficiency. This application note provides practical insights into how ground plane size, shape, and placement influence antenna behavior, offering guidance for optimal integration in real-world designs. Simulation results and measurement data are included to illustrate key effects and support design decisions.

1. Revision history

| Manual version | FW version | HW version | Notes | Date |
|----------------|------------|------------|--|---------------|
| 1.0 | 1.0.0 | 1.0 | <ul style="list-style-type: none"> Initial release | January 2022 |
| 1.1 | 1.0.0 | 1.0 | <ul style="list-style-type: none"> Added radiation characteristics in chapter 20.3 Updated layout in chapter 20.2 Updated meta data | May 2023 |
| 1.2 | 1.0.0 | 1.0 | <ul style="list-style-type: none"> Added certificates in addition to the required compliance statements in chapter Regulatory compliance information. | June 2023 |
| 1.3 | 1.0.0 | 1.0 | <ul style="list-style-type: none"> Added new radio certification for India in chapter 28.12 | August 2023 |
| 1.4 | 1.1.0 | 1.0 | <ul style="list-style-type: none"> Added chapter Known issues and known issue KI-001 Updated chapter Transparent mode Added information concerning new firmware version Improved description of direct test mode in chapter 7.10 | December 2023 |
| 1.5 | 1.1.0 | 1.0 | <ul style="list-style-type: none"> Added NCC compliance statement for Taiwan chapter 28.13 | February 2024 |
| 1.6 | 1.1.0 | 1.0 | <ul style="list-style-type: none"> Added chapter Important notice UKCA and Product testing | October 2024 |
| 1.7 | 1.1.0 | 1.0 | <ul style="list-style-type: none"> Added chapter Certification of the end device and Hardware history | October 2025 |
| 1.8 | 1.1.0 | 1.0 | <ul style="list-style-type: none"> Update CE declaration in chapter Regulatory compliance information. | January 2026 |

★ For firmware history see chapter Firmware history

2. Abbreviations

| Abbreviation | Name | Description |
|------------------|--|--|
| ADC | Analog to Digital Converter | |
| ASCII | American Standard Code For Information Interchange | |
| BER | Bit Error Rate | |
| BTMAC | | Bluetooth® conform MAC address of the module used on the RF-interface |
| CS | Checksum | Byte wise XOR combination of the preceding fields |
| DC | Direct Current | |
| DTM | Direct Test Mode | Mode to test Bluetooth® specific RF settings |
| ESD | Electrostatic Discharge | |
| EV (Board) | Evaluation (Board) | Proteus-e populated on motherboard with USB interface for test and evaluation purposes |
| FCC | Federal Communications Commission | |
| FEC | Forward error correction | Technique to correct received erroneous radio messages |
| FW | Firmware | |
| GAP | Generic Access Profile | The GAP provides a basic level of functionality that all Bluetooth® devices must implement |
| GND | Ground | Ground signal level that corresponds to 0 V |
| GPIO | General Purpose Input/Output | |
| HFXO | High-Frequency Crystal Oscillator | |
| HIGH | | High signal level that corresponds to VDD |
| IC | Integrated Circuit | |
| I/O | Input/Output | Pinout description |
| I ² C | Inter-Integrated Circuit | |
| Bluetooth LE | Bluetooth Low Energy | |
| LED | Light Emitting Diode | |
| LDO | Low Dropout | Low dropout voltage regulator |
| LOW | | Low signal level that corresponds to 0 V |

| | | |
|---------------|---|--|
| LSB | Least Significant Bit/Byte | |
| MAC | | MAC address of the module |
| MPS | Maximum Payload Size | The maximum size of the payload, that can be transmitted/received using one Bluetooth® LE transaction |
| MSB | Most Significant Bit/Byte | |
| MTU | Maximum transmission unit | Maximum packet size of the Bluetooth® connection |
| OS | Operating System | |
| Payload | | The intended message in a frame / package |
| PCB | Printed Circuit Board | |
| PCN | Product Change Notification | |
| PER | Packet Error Rate | |
| RAM | Random Access Memory | |
| 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 |
| Bluetooth SIG | Bluetooth Special Interest Group | |
| Soft device | | Operating system used by the nRF52 chip |
| SPI | Serial Peripheral Interface | Allows the serial communication with the module |
| SMA | SubMiniature version A | |
| SPP | Serial Port Profile | |
| SWD | Serial Wire Debug | Flash und debug interface |
| UART | Universal Asynchronous Receiver Transmitter | Allows the serial communication with the module |
| User settings | | Settings to configure the module. Any relation to a specific entry in the user settings is marked in a special font and can be found in chapter 10 |
| UUID | Universally Unique Identifier | |
| [HEX] 0xhh | Hexadecimal | All numbers beginning with 0x are hexadecimal numbers. All other numbers are decimal, unless stated otherwise |
| VDD | Voltage Drain Drain | Supply voltage |

| | | |
|-------|----------------------------------|--|
| VSWR | Voltage Standing Wave Ratio | |
| WLCSP | Wafer Level Chip Scale Packaging | |

3. Introduction

3.1. Operational description

The Proteus-e is a radio module for wireless communication between devices such as control systems, remote controls, sensors etc. On the basis of Bluetooth® LE 5.1 [1], it offers fast and secure data transmission of data packages between two parties (point to point topology). Within these two parties, Proteus-e implements the Bluetooth LE role "peripheral". The other device (e.g. a Proteus-III or a smart phone) must implement Bluetooth® LE role "central" and the Bluetooth® LE profiles and characteristics Proteus-e is offering.

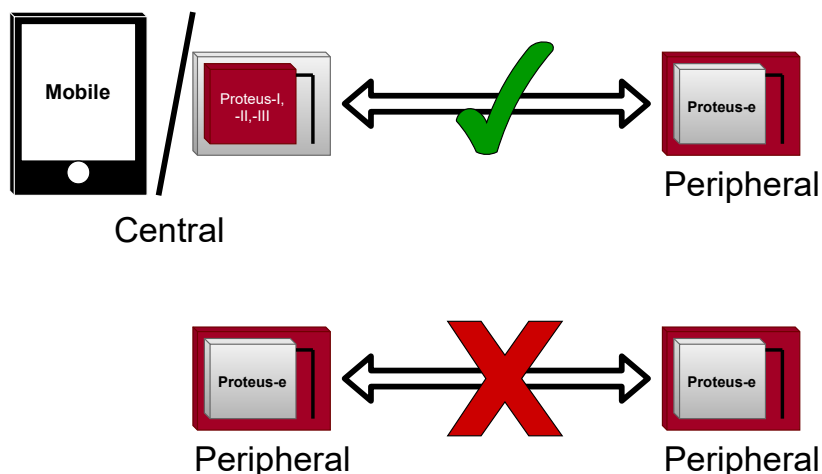


Figure 1: Compatible central devices implementing the Bluetooth® LE profiles of Proteus-e

The Bluetooth® LE standard itself offers very high configurability, making the module suitable for a wide range of applications. For example, the module can be configured to communicate with external sensors, can use security configurations and can be optimized for low power consumption. To fulfill the specific requirements of the application, a tailored firmware can be developed on the basis of the Proteus-e hardware (see chapter 15).

Ultra small dimensions of 7 x 9 mm including a strongly miniaturized PCB antenna make the Proteus-e ideal for small form factor design. It is possible to connect an external antenna, in case higher radio ranges are required.

A serial interface (UART) is available for communication with the host system.

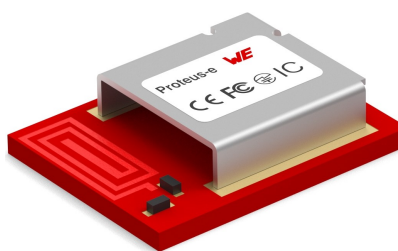


Figure 2: Proteus-e

3.1.1. Key features

The Proteus-e offers the following key features:

SPP-like connection-based secured data transmission: The Proteus-e firmware implements an SPP-like Bluetooth® LE profile that allows the bidirectional data transmission between a Proteus-e module and one other Bluetooth® LE device implementing the SPP-like profile. Secured connections allow the transmission of encrypted and authenticated data.



More than one connection at a time is not supported.

Remote GPIOs: The Proteus-e firmware allows to switch free module GPIOs via remote control. More information can be found in chapter 13.

Advanced customization capabilities: The configurable UUID and appearance of the Bluetooth® LE profile allow personalization of the Proteus-e to fuse with the user's end product. For more information see chapter 14.

Custom advertising data: Furthermore, the complete content of the advertising and scan response packet can be customer defined. With this, any beacon function can be realized. For more information see chapter 7.7.

Low power position sensing solutions: The TX power of any Proteus-e is always transmitted during advertising. With this, distance estimation and position sensing solutions can be realized conveniently.

Fast serial interface: The Proteus-e offers a UART-interface to communicate with a host using a user-defined baud rate and a simple command interface. Furthermore, the firmware provides the so called "transparent mode" (see chapter 12), which is formerly known as "peripheral only mode" in Proteus-I,-II and -III. This mode offers a transparent UART interface such that an easy adaption of already existing custom hardware with the Bluetooth® LE interface is enabled.

Modern microprocessor provided by Nordic Semiconductor nRF52 series: The heart of the Proteus-e is a Bluetooth® LE chip of the nRF52 series offering high performance values combined with low power consumption. It is a 64 MHz ARM Cortex-M4 CPU with 192kB flash + 24kB RAM and up to 4 dBm output power.

Bluetooth® 5 stack: The Bluetooth® 5 stack enables fast and energy efficient data transmission using state-of-the-art technology of Nordic Semiconductors.

Flexible wired interfacing: The Proteus-e is equipped with extra pins suited for custom device/sensor connection. With help of these, a tailored firmware can be developed which is optimized to the customer's needs. The pins can be configured to various functions such as UART, SPI, I2C, ADC and GPIO.

Additional Bluetooth® 5 radio modes: The Proteus-e provides the advanced radio modes 2 MBit mode for faster data transmission. For more information, see chapter 7.11.

3.1.2. Connectivity

The Bluetooth® LE standard allows to setup a network with various Bluetooth® LE devices from different manufacturers. To be able to communicate with Proteus-e devices, the SPP-like profile must be known and implemented by all network participants.

The advanced developer guide of Proteus-e (application note ANR024 [2]) contains the design data of the SPP-like profile, to implement it for example in smart phone apps.

3.2. Block diagram

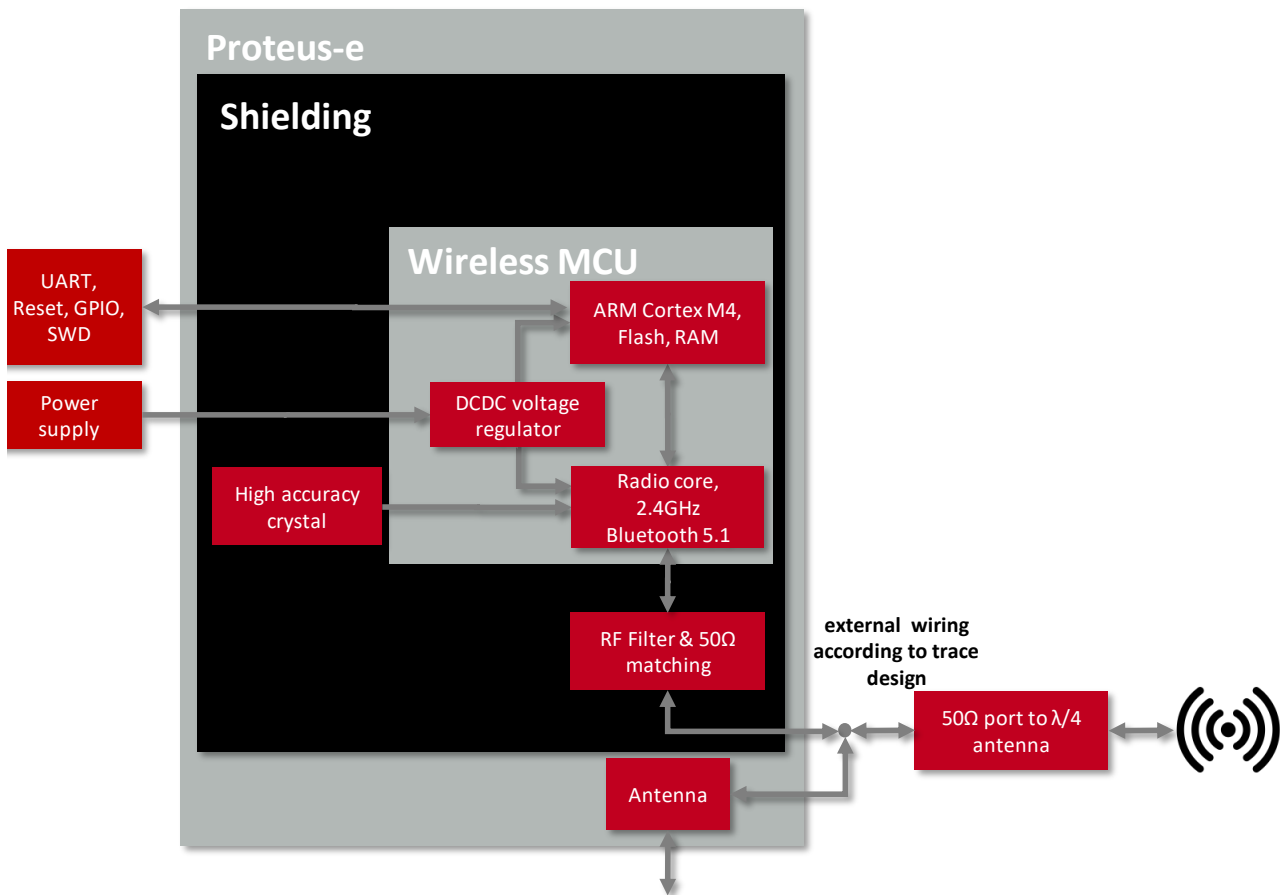


Figure 3: Block diagram of the module

3.3. Ordering information

| WE order code | Description |
|---------------|---|
| 2612011024000 | Proteus-e Bluetooth® LE Module, Tape & Reel |
| 2612019024001 | Proteus-e EV-Board |

Table 3: Ordering information

4. Electrical specifications

As not otherwise stated measured on the EV-Board of the Proteus-e with $T = 25\text{ }^{\circ}\text{C}$, $V_{DD}=3\text{V}$, $f=2.44\text{GHz}$, internal DC-DC converter in use.

4.1. Operating conditions

| Parameter | Min. | Typ. | Max. | Unit |
|---|------------------|------|------|--------------------|
| Ambient temperature | -40 | 25 | 85 | $^{\circ}\text{C}$ |
| Supply voltage (V_{DD}) | 1.8 ¹ | 3 | 3.6 | V |
| Supply rise time (0V to $\geq 1.8\text{ V}$) | | | 60 | ms |

Table 4: Operating conditions



The on-chip power-on reset circuitry may not function properly for rise times longer than the specified maximum. A step increase in supply voltage of 300 mV or more, with rise time of 300 ms or less, within the valid supply range, may result in a system reset.



An instable supply voltage may significantly decrease the radio performance and stability.

4.2. Absolute maximum ratings

| Parameter | Min. | Max. | Unit |
|---|--------|----------------|--------------------|
| Supply voltage (V_{DD}) | -0.3 | +3.9 | V |
| Voltage on any digital pin ($V_{DD} < 3.6\text{ V}$) | -0.3 | $V_{DD} + 0.3$ | V |
| Voltage on any digital pin ($V_{DD} \geq 3.6\text{ V}$) | -0.3 | 3.9 | V |
| Input RF level | | 10 | dBm |
| Flash endurance | 10 000 | | Write/erase cycles |

Table 5: Absolute maximum ratings

¹Power fail comparator is set to 1.8 V ($\pm 5\%$) to avoid flash fail due to voltage drop.

4.3. Power consumption

4.3.1. Static

| Parameter | Power | Test conditions | Typ. | Unit |
|------------------------|----------------|--|------|------|
| TX Current consumption | RF_TXPower = 4 | Transmitter only, DC/DC converter enabled, nRF52 data sheet, CPU current not included | 8 | mA |
| | | Full module current consumption, DC/DC converter enabled, measured | 9.3 | mA |
| | RF_TXPower = 0 | Transmitter only, DC/DC converter enabled, nRF52 data sheet, CPU current not included | 5.8 | mA |
| | | Full module current consumption, DC/DC converter enabled, measured | 7.1 | mA |

Table 6: Current consumption - transmitting

| Parameter | Test conditions | Typ. | Unit |
|------------------------|--|------|------|
| RX Current consumption | Receiver only, DC/DC converter enabled nRF52 data sheet, CPU current not included | 6.1 | mA |
| | Full module current consumption, DC/DC converter enabled, measured | 6.8 | mA |

Table 7: Current consumption - receiving

| Parameter | Test conditions | Typ. | Unit |
|---|--|------|------|
| Current consumption | Sleep (system off mode) | 0.3 | μA |
| Current consumption reduction through CMD_UARTDISABLE_REQ | Initial state: Bluetooth® LE connected or idle | 250 | μA |

Table 8: Current consumption - low power



Due to the Bluetooth® LE time slot operation, the real operating currents are reduced significantly and depend on the user selectable advertising and connection interval settings.



Sleep current is significantly increasing for temperatures above 30 °C.

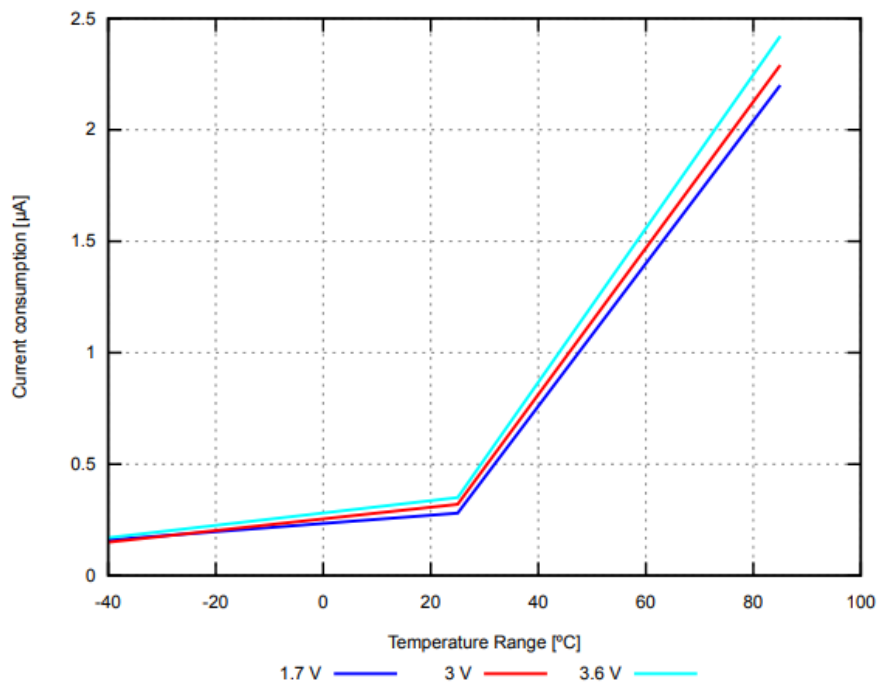


Figure 4: Sleep current (no RAM retention, wake on reset) over operating temperature range

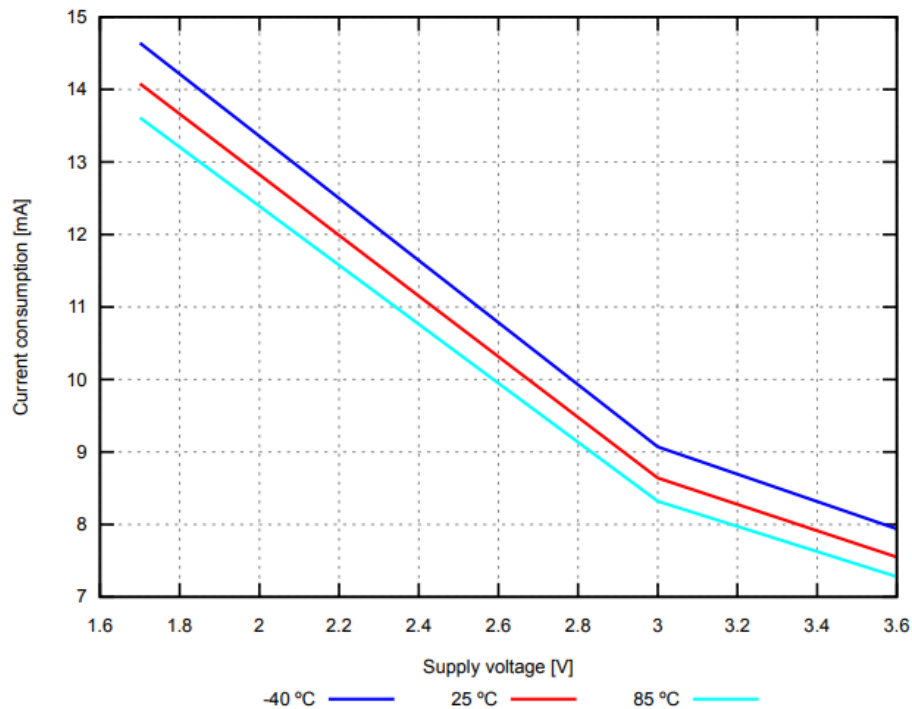


Figure 5: Radio transmitting @ 4 dBm output power, 1 Mbps Bluetooth® LE mode, Clock = HFXO, Regulator = DC/DC (typical values)

4.3.2. Dynamic

Besides the static TX, RX, idle and sleep current, the average current is of interest. Here an example for a typical behavior of a peripheral device in advertising mode (see Figure 6). Currents and state durations are dependent on the configuration of the module. In this state the module transmits the advertising packets on the three advertising channels.

Nordic Semiconductor provides an online tool calculating the average current of a Bluetooth® connection. It can be accessed at <https://devzone.nordicsemi.com/power/>.

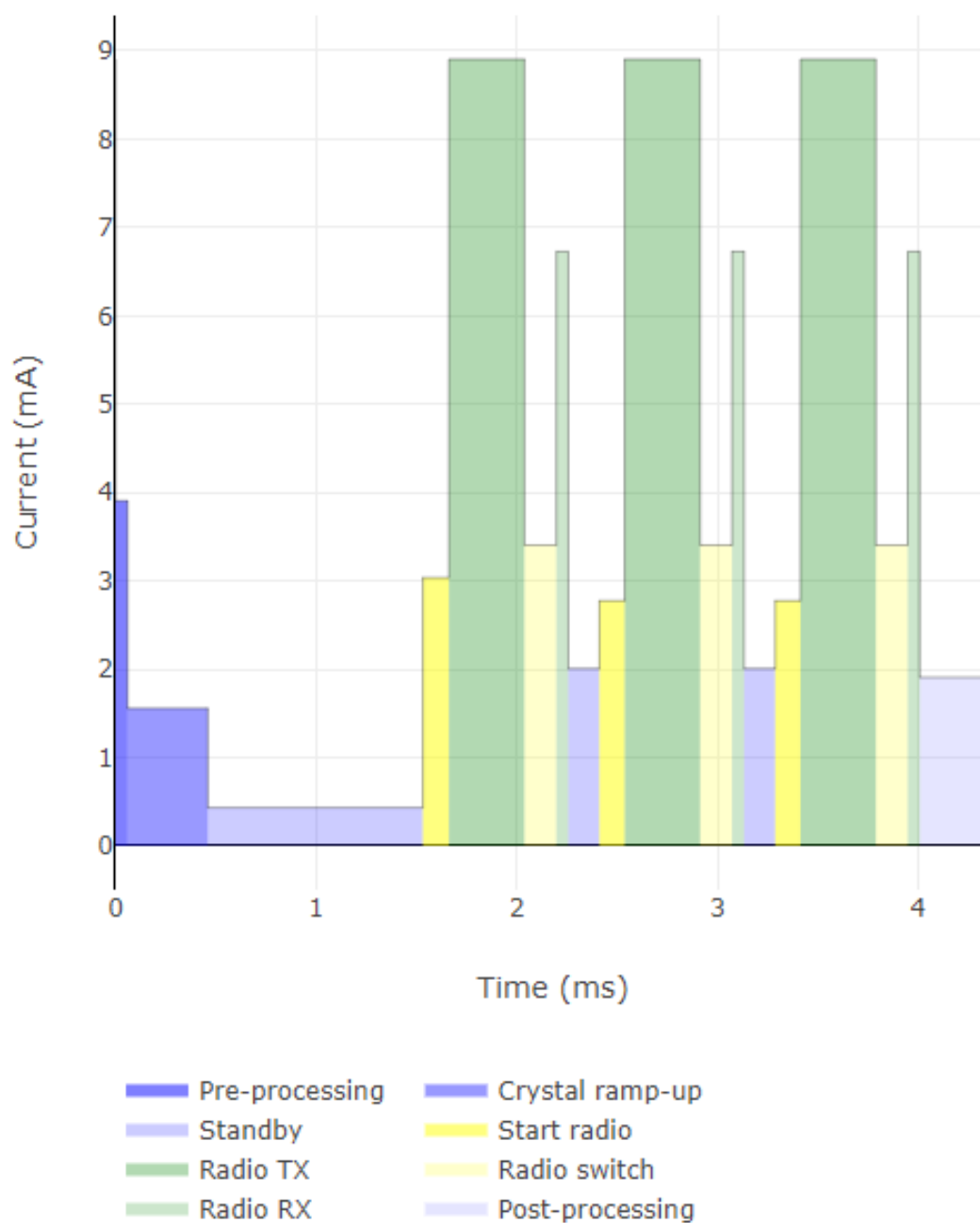


Figure 6: Current consumption calculation in advertising mode with 40 ms advertising interval with 4 dBm output power and 31 Bytes radio data, UART disabled

4.4. Radio characteristics

| Parameter | Min. | Max. | Unit |
|---|------|------|------|
| RSSI accuracy valid range (± 2 dB) | -90 | -20 | dBm |

Table 9: RSSI accuracy

| Parameter | Typ. | Unit |
|-----------------------------------|------|---------|
| Enable TX or RX delay | 140 | μ s |
| Enable TX or RX delay (fast mode) | 40 | μ s |
| Disable TX delay | 6 | μ s |
| Disable RX delay | 0 | μ s |

Table 10: Timing

| Parameter | Test conditions | Typ. | Unit |
|-------------------|--------------------------------------|------|------|
| Output power | RF_TXPower = 4, conducted | +4 | dBm |
| | RF_TXPower = 4, radiated | -4 | dBm |
| Input sensitivity | Conducted, BER=1E-3, 1Mbps | -93 | dBm |
| | Conducted, BER=1E-3, 1Mbps, LDO mode | -97 | dBm |
| | Radiated, BER=1E-3, 1Mbps | -85 | dBm |

Table 11: Transmit and receive power

All transmit and receive power levels are measured on the EV-Board. The values already include losses of transitions from module to motherboard to SMA or modules PCB antenna. They are realistic values for the end application.

Sensitivity in the table above is stated for the common used bit error rate of 0.1% (BER 1E-3). In the table below the sensitivity is stated for a packet error rate of 1% with a payload length of 38 byte at different data rates. The PER 1% is a harder criteria resulting in 2 dBm less sensitivity.

| Parameter | Test conditions | Typ. | Unit |
|-------------------|-------------------|------|------|
| Input sensitivity | 1Mbit Phy, PER 1% | -91 | dBm |
| | 2Mbit Phy, PER 1% | -88 | dBm |

Table 12: Sensitivity at different data rates

4.5. Pin characteristics

When configured as digital pin output "standard drive" is used in the Proteus-e firmware.

| Parameter | Min. | Typ. | Max. | Unit |
|--|------------------|------|------------------|------------|
| Input high voltage | $0.7 \times VDD$ | | VDD | V |
| Input low voltage | VSS | | $0.3 \times VDD$ | V |
| Current at VSS+0.4 V, output set low, standard drive , $VDD \geq 1.7V$ | 1 | 2 | 4 | mA |
| Current at VSS+0.4 V, output set low, high drive, $VDD \geq 2.7 V$ | 6 | 10 | 15 | mA |
| Current at VSS+0.4 V, output set low, high drive, $VDD \geq 1.7 V$ | 3 | | | mA |
| Current at VDD-0.4 V, output set high, standard drive , $VDD \geq 1.7V$ | 1 | 2 | 4 | mA |
| Current at VDD-0.4 V, output set high, high drive, $VDD \geq 2.7 V$ | 6 | 9 | 14 | mA |
| Current at VDD-0.4 V, output set high, high drive, $VDD \geq 1.7 V$ | 3 | | | mA |
| Internal pull-up resistance | 11 | 13 | 16 | k Ω |
| Internal pull-down resistance | 11 | 13 | 16 | k Ω |

Table 13: Pin characteristics

5. Pinout

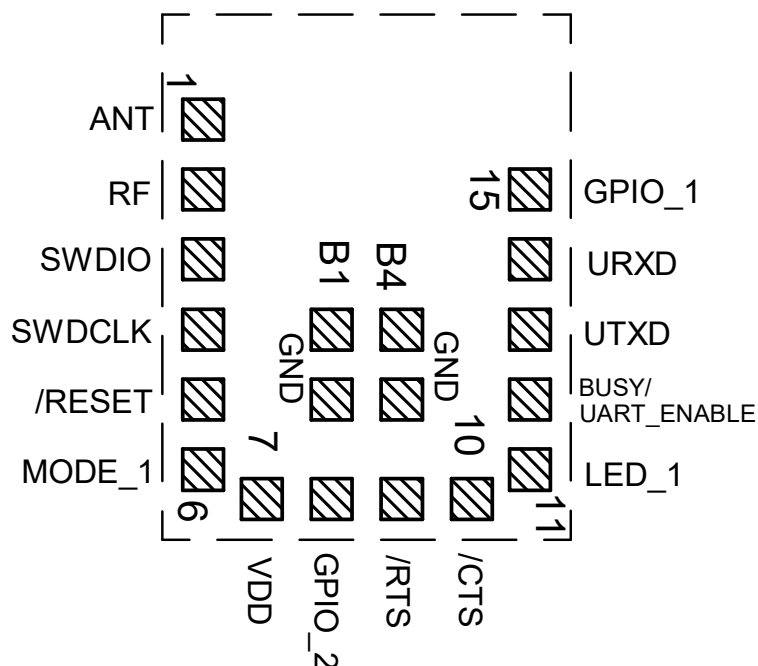


Figure 7: Pinout (top view)

| No | μC Pin | Designation | I/O | Description |
|----|--------|---------------|-------|--|
| 1 | | <i>ANT</i> | I/O | 50 Ω RF connection to PCB antenna. (see chapter 6.2) |
| 2 | | <i>RF</i> | I/O | 50 Ω RF connection through radio front end to transceiver part of chipset. (see chapter 6.2) |
| 3 | | <i>SWDIO</i> | Input | Serial wire input/output (SWD Interface). Uses internal pull up resistor. Do not connect if not needed. |
| 4 | | <i>SWDCLK</i> | Input | Serial wire clock (SWD Interface). Uses internal pull down resistor. Do not connect if not needed. |
| 5 | P0.21 | <i>/RESET</i> | Input | Reset pin. A LOW signal resets the module. Uses internal pull up resistor. |
| 6 | P0.12 | <i>MODE_1</i> | Input | Operation mode pin with internal pull down resistor ¹ during start-up. LOW level or open: Command mode. HIGH level: Transparent mode. Do not connect if not needed. |

| | | | | |
|----|------------------------|------------------------------|-------------------|---|
| 7 | | <i>VDD</i> | Supply | Supply voltage |
| 8 | P0.05 | <i>GPIO_2</i> | GPIO | Pin for remote GPIO access. Do not connect, if not needed. |
| 9 | P0.04 | <i>/RTS</i> | Output | Static low, if flow control is disabled. <i>/RTS</i> signal, otherwise if UART is enabled ³ . Do not connect if not needed. |
| 10 | P0.14 | <i>/CTS</i> | Input | Internal pull down ¹ , if flow control is disabled. <i>/CTS</i> signal, otherwise if UART is enabled ³ . Do not connect if not needed. |
| 11 | P0.00/XL1 ² | <i>LED_1</i> | Output | Indicates the module state (active HIGH). Do not connect if not needed. |
| 12 | P0.01/XL2 ² | <i>BUSY/ UART_ENABLE</i> | Input / Output | Shared pin. In Transparent mode (see chapter 12.3.1) this is an output pin, that indicates if the module is busy with data transmission. In Command mode this is an input pin with internal pull-up, that is used to wake up the UART, in case the UART has been disabled before using the <i>CMD_UARTDISABLE_REQ</i> command. To do so, apply a falling edge, holding the line LOW for at least 12 ms before applying a rising edge and holding it HIGH. Do not connect if not needed. |
| 13 | P0.16 | <i>UTXD</i> | Output | UART (Transmission), if UART is enabled ³ . |
| 14 | P0.18 | <i>URXD</i> | Input | UART (Reception) with internal pull up resistor ¹ , if UART is enabled ³ . |
| 15 | P0.20 | <i>GPIO_1</i> | GPIO | Pin for remote GPIO access. Do not connect, if not needed. |
| B1 | | <i>GND</i> | Supply | Ground |
| B2 | | <i>GND</i> | Supply | Ground |
| B3 | | <i>GND</i> | Supply | Ground |
| B4 | | <i>GND</i> | Supply | Ground |

Table 14: Pinout

¹Internal pull ups or pull downs are configured at start-up by the firmware installed in the SoC. The pull up on the */RESET* pin cannot be disabled by firmware.

²Pins available to connect an external crystal in custom firmware. The standard firmware of Proteus-e does not implement this function.

³If UART is disabled, this pin is input floating. UART can be disabled using the *CMD_UARTDISABLE_REQ* command in command mode, or closing the Bluetooth® LE connection in transparent mode.

6. Quick start

6.1. Minimal pin connections

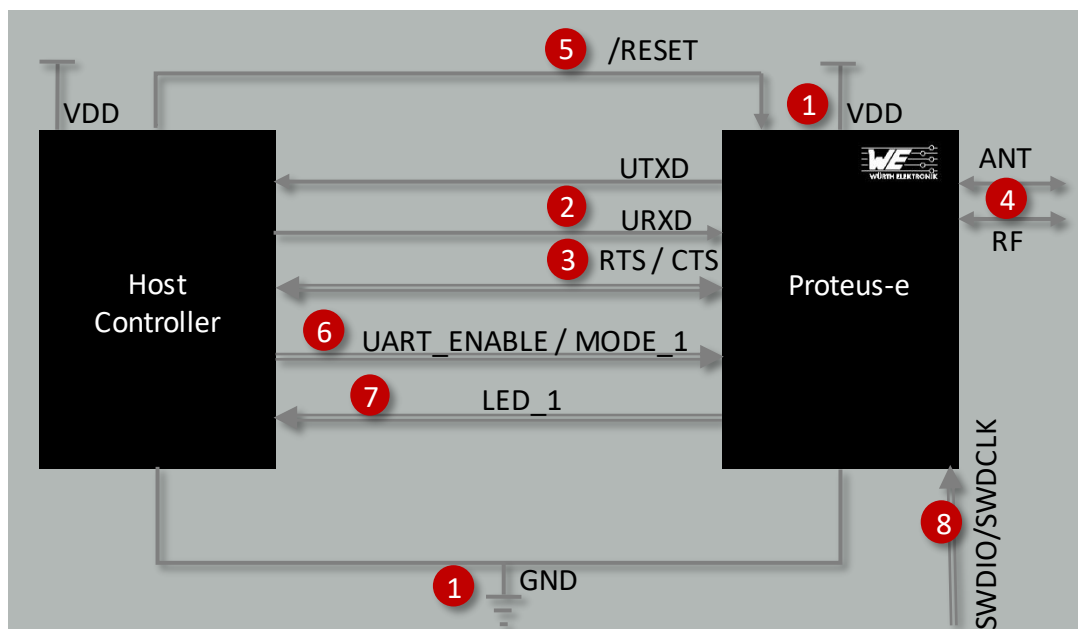


Figure 8: Minimal pin connections

The above image shows the steps to be performed to integrate the Proteus-e into a custom end device.

1. Supply voltage and ground
Connect the *VDD* and *GND* pins to supply the radio module with power.
2. UART serial interface to the host
Connect the UART pins *UTXD* and *URXD* to the host to control the module via host.
3. UART flow control
In case of UART baudrates higher than 115.2 kbaud, the UART flow control is activated automatically. For lower data rates, the flow control is inactive per default. If activated the */RTS* and */CTS* pins must be connected to the host controller.
4. Antenna connection
The antenna configuration must be performed. See chapter 6.2.
5. Reset
Connect the */RESET* pin to the host to allow a hard reset of the module.
6. (Optional) Enable/Disable UART and mode selection

- Connect the *MODE_1* pin to the host controller to switch between command and transparent mode.
- In case the transparent mode or the feature of switching off the UART in command mode shall be used, connect the shared *BUSY/UART_ENABLE* pin to the host controller.

7. (Optional) Status indication

Connect the *LED_1* pin to the host controller to allow easy indication of the status.

8. (Optional) Flash and debug interface

In case of custom firmware development, it is recommended to additionally have the pins *SWDIO* and *SWDCLK* accessible in order to support a fail-safe update of firmware. A standard socket on the customer's PCB for connecting a flash adapter can be useful for debugging purposes (e.g. a JTAG 2*10 pin header with 2.54 mm pin-to-pin distance).

If the module has to be connected to a PC, a converter (TTL to RS-232 or TTL to USB) has to be used. See chapter *Pinout* for details on all pins. Please refer to the Proteus-e EV-Board schemes for a *Reference design*.



The logic level of the module is based on 3V. A 5 V logic level shall not be connected directly to the module.

6.2. Antenna connection

Proteus-e's smart antenna configuration allows the user to choose between two antenna options. For more information on how to use them and what to consider for certification aspects is described in chapter 20:

6.2.1. On-board PCB antenna

The Proteus-e has an on-board PCB antenna optimized for strong miniaturization operating in the 2.4 GHz frequency band. To use this integrated antenna, it has to be connected to the radio chip by connecting the pins *RF* and *ANT*.

6.2.2. External antenna

For applications that use an external antenna, the Proteus-e provides a 50 Ω RF signal on pin *RF* of the module. In this configuration, pin *ANT* of the module has to be left open and pin *RF* has to be connected to the external antenna via 50 Ω feed line.



The use cases for the integrated antenna are miniaturization and re-use of module certifications for the end-application. An external antenna is normally used to increase the achievable radio range, at the cost of more space needed in the device/application. Also, an external antenna could be needed to fit the specific application environment of the module, for example when a metal housing is used.

6.3. Power up

After powering the module the */RESET* pin shall be held for another Δt of 1 ms after the *VDD* is stable to ensure a safe start-up. The module will send a *CMD_GETSTATE_CNF* (0x02 41 02 00 01 01 41) to indicate "ready for operation" after the */RESET* pin was released.



Applying a reset (e.g. a host temporarily pulling the */RESET* pin down for at least 1 ms and releasing it again) after the *VDD* is stable will also be sufficient.

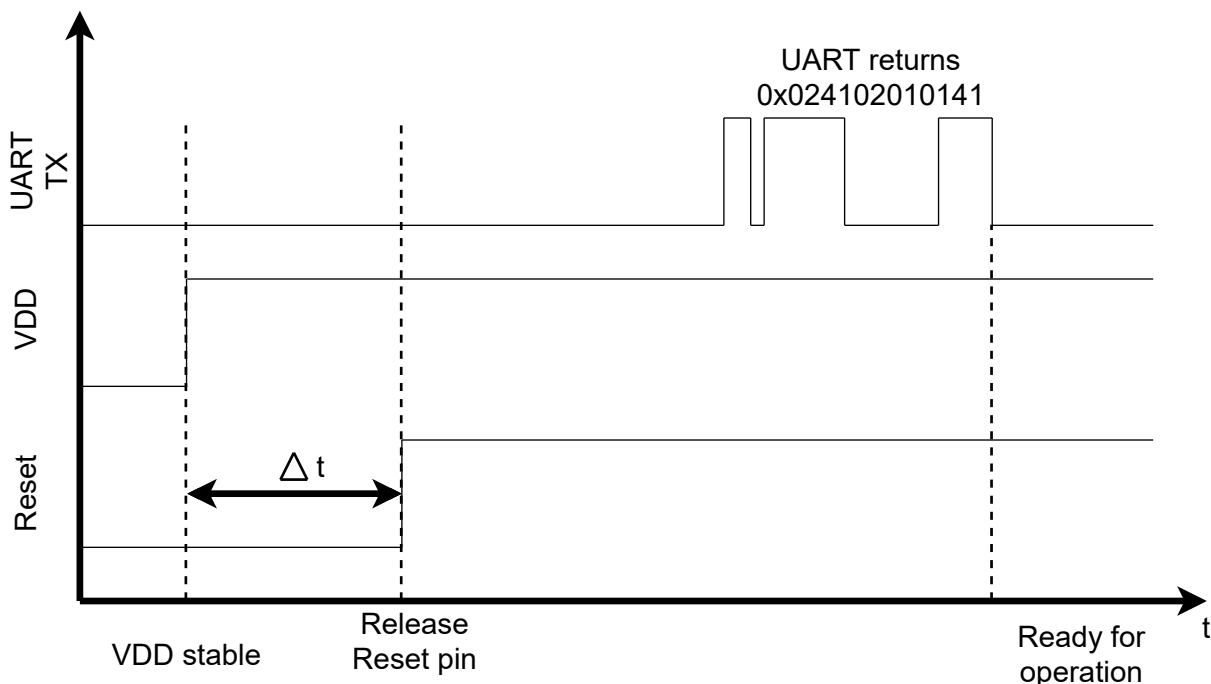


Figure 9: Power up

6.4. Quickstart example

The quick start example in this chapter demonstrates how the UART communication with the Proteus-e works in command mode. In case the host sends a request message (i.e. CMD_GET_REQ) to the radio module, it responds with a confirmation message (i.e. CMD_GET_CNF).

In this example several parameters of the radio module are requested and the device name is configured.



Quick start examples demonstrating the connection setup and data transmission via radio can be found in the application note ANR025 [3].



The below commands are in hexadecimal notation. The arrow in the left column indicates whether it's a message from host to radio module or vice versa. A request command is always sent from host to module (\Rightarrow). An indication, confirmation or response message is always sent from module to host (\Leftarrow).

1. Power-up the module and make its UART accessible by the host (115200 Baud, 8n1). After the power-up or after reset the following sequence is sent from the module to the host.

| Info | Module |
|--|----------------------|
| \Leftarrow Response CMD_GETSTATE_CNF: Module started in ACTION_IDLE mode | 02 41 02 00 01 01 41 |

2. Request the FS_BTMAC and FS_SerialNumber of the module.

| Info | Module |
|---|-------------------------------------|
| \Rightarrow Request CMD_GET_REQ with settings index 4 | 02 10 01 00 04 17 |
| \Leftarrow Response CMD_GET_CNF: FS_BTMAC is 0x55 0x00 0x00 0xDA 0x18 0x00 | 02 50 07 00 00 55 00 00 DA 18 00 C2 |
| \Rightarrow Request CMD_GET_REQ with settings index 16 | 02 10 01 00 10 03 |
| \Leftarrow Response CMD_GET_CNF: FS_SerialNumber is 0x55 0x00 0x00 0xDA 0x18 0x00 | 02 50 04 00 00 55 00 00 57 |

3. Set the RF_DeviceName of the module to "Hello World" (0x48 65 6C 6C 6F 20 57 6F 72 6C 64).

| Info | Module |
|--|--|
| ⇒ Request CMD_SET_REQ with settings index 2 | 02 11 0C 00 02 48 65 6C 6C 6F 20 57 6F 72 6C 64 3D |
| ⇐ Response CMD_SET_CNF: Successfully modified the setting | 02 51 01 00 00 52 |
| ⇐ Response CMD_GETSTATE_CNF: Module re-started in ACTION_IDLE mode | 02 41 02 00 01 01 41 |

7. Functional description

7.1. Operation modes

The Proteus-e module acts as a slave and can be fully controlled by an external host. The Proteus-e supports the following operating modes:

- The **command mode**, where the Proteus-e can be controlled by the host controller via commands. The command mode allows to use all functions of the radio module. Functions, like data transmission or configuration tasks, can be triggered by predefined commands (see chapter 9) that are sent as telegrams over the UART interface.
- The **transparent mode** (see chapter Transparent mode) provides a transparent UART interface. Data transmission can be done by the host without using any commands.

7.2. Radio module states

The Proteus-e can operate in different states. Depending on the active state several commands of the command interface (see chapter 9) are permitted to modify the state, configure the module or transmit data over the radio interface. An overview of the different states and the corresponding allowed commands can be found in Figure 10.

When the Proteus-e is powered up, it starts in ACTION_IDLE state. In this state the module advertises, such that other devices in range can detect it and connect to it.

The ACTION_CONNECTED state can be entered by getting a connection request from another Bluetooth® LE device operating in central role. In this case, it stops advertising and data can be transmitted and received to/from the connected Bluetooth® LE device. This state remains active until the module disconnects itself, or a disconnection request from the connected remote device is received.

When disconnecting, the module goes to ACTION_IDLE state and starts advertising again to be ready for the next connection setup.

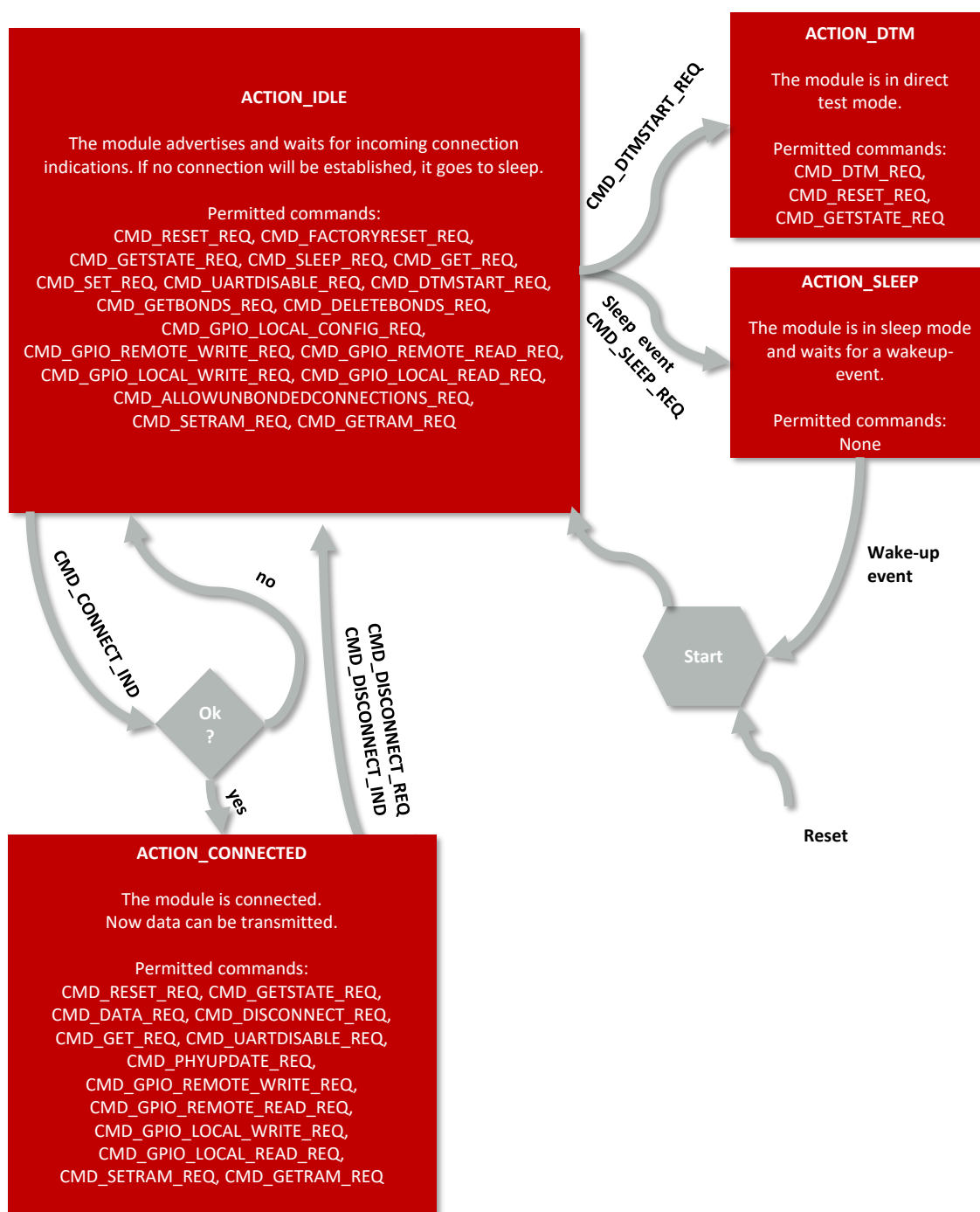


Figure 10: State overview

7.3. State indication using the LED pins

The pin *LED_1* of the Proteus-e can be used to determine the module state. The states described in Figure 10 result in the following pin behavior. The pins on the Proteus-e are active HIGH.

| State | <i>LED_1</i> |
|------------------|---|
| ACTION_IDLE | Blinking (On for 200 ms, Off for 2800 ms) |
| ACTION_CONNECTED | Blinking (On for 250 ms, Off for 250 ms) |
| | Static On (as soon as the channel was opened successfully, see CMD_CHANNELOPEN_RSP) |
| ACTION_SLEEP | Off |
| ACTION_DTM | Off |

Table 15: LED behavior of the Proteus-e

7.4. Sleep mode

Especially for battery-powered devices the ACTION_SLEEP mode (system-off mode) supports very low power consumption. It can be entered by sending the command CMD_SLEEP_REQ to the module. As response, the module will send a CMD_SLEEP_CNF and then enter the ACTION_SLEEP mode.

In ACTION_SLEEP mode the UART is disabled. Thus the module will not receive or transmit any data. To prevent leakage current, the host shall not pull the *URXD* to LOW level, as the module has an internal pull-up resistor enabled on this pin. The GPIO pins *GPIO_1* and *GPIO_2* are set to input without pull resistor during the sleep period.

The ACTION_SLEEP mode can be entered only if the module is in state ACTION_IDLE, that means no peer device is connected via radio.

To leave the ACTION_SLEEP mode and enter ACTION_IDLE state again, the module has to be woken up by a pin reset; apply a LOW signal to the *RESET* pin for at least 1 ms before releasing the signal back to HIGH. The module then restarts completely, so that all volatile settings are set to default. A CMD_GETSTATE_CNF will be sent when the module is ready for operation again.

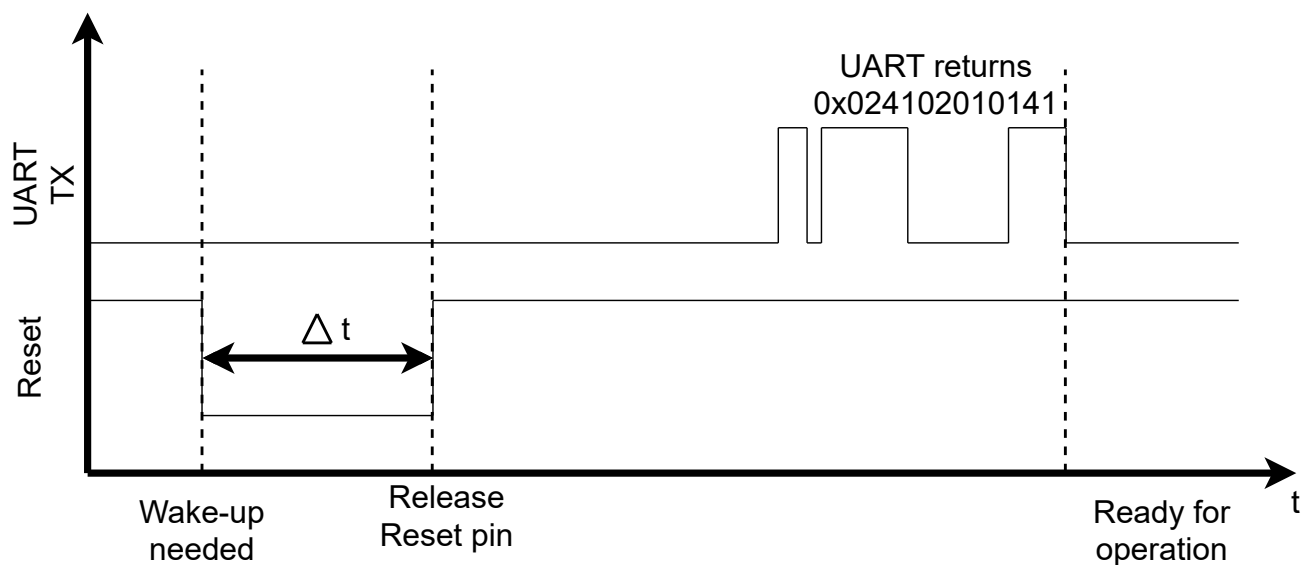


Figure 11: Power up

7.5. Identification of a Proteus-e device on the radio

The Proteus-e can be identified on the radio interface by its 6 Bytes long Bluetooth®-conform MAC address FS_BTMAC, which is part of the data package sent during advertising in ACTION_IDLE mode.

To simplify the identification of Proteus-e devices on the Bluetooth® LE interface a short user-defined name (see user setting RF_DeviceName) can be given to the module, which is also part of the advertising packet.



The FS_BTMAC consists of the Würth Elektronik eiSos MAC ID 0x0018DA followed by the module's serial number FS_SerialNumber.

7.6. Connection based data transmission

In the Bluetooth® LE standard the data transmission typically is connection based. A connection between two devices can be secured or unsecured (see user setting RF_SecFlags). The connection setup is triggered by the central device and consists of several steps, that must be run sequentially:

1. Physical connection establishment
A physical connection has to be established first. Therefore, a central device (e.g. smart phone) has to connect to the Proteus-e which runs as peripheral.
2. Pairing process (optional, in case the user setting RF_SecFlags has been set)
The authentication and exchange of encryption information is part of the pairing process. The central device must request at least the same security level configured in Proteus-e to access the characteristics of the Proteus-e.



In case the Proteus-e has enabled a security mode, but the central device goes on with the next steps without placing the pairing 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 configured in the Proteus-e.

3. Exchange of the maximum transmission unit (MTU) (optional)

The maximum transmission unit can be increased to allow the transmission of larger data packets. The Proteus-e allows an MTU of up to 247 bytes, which results in a maximum payload size (MPS) of 243 bytes. Not selecting a higher MTU will use the Bluetooth® LE 4.0 default MTU which results in a MPS of 19 bytes, but will be compatible to pre Bluetooth® LE 4.2 devices and post Bluetooth® LE 4.2 devices not implementing a larger MTU than required by the Bluetooth® standard.

4. Discover the characteristics of the Proteus-e SPP-like profile

The characteristics offered by the Proteus-e have to be discovered by the central.

5. Notification enable

To transmit data from the peripheral to the central, the central must enable the notifications on the peripheral's characteristics. After this step, the channel is open and data transmission can start. In case of transparent mode, the UART is enabled at this time.

As soon as the connection has been established data can be transmitted in both directions.

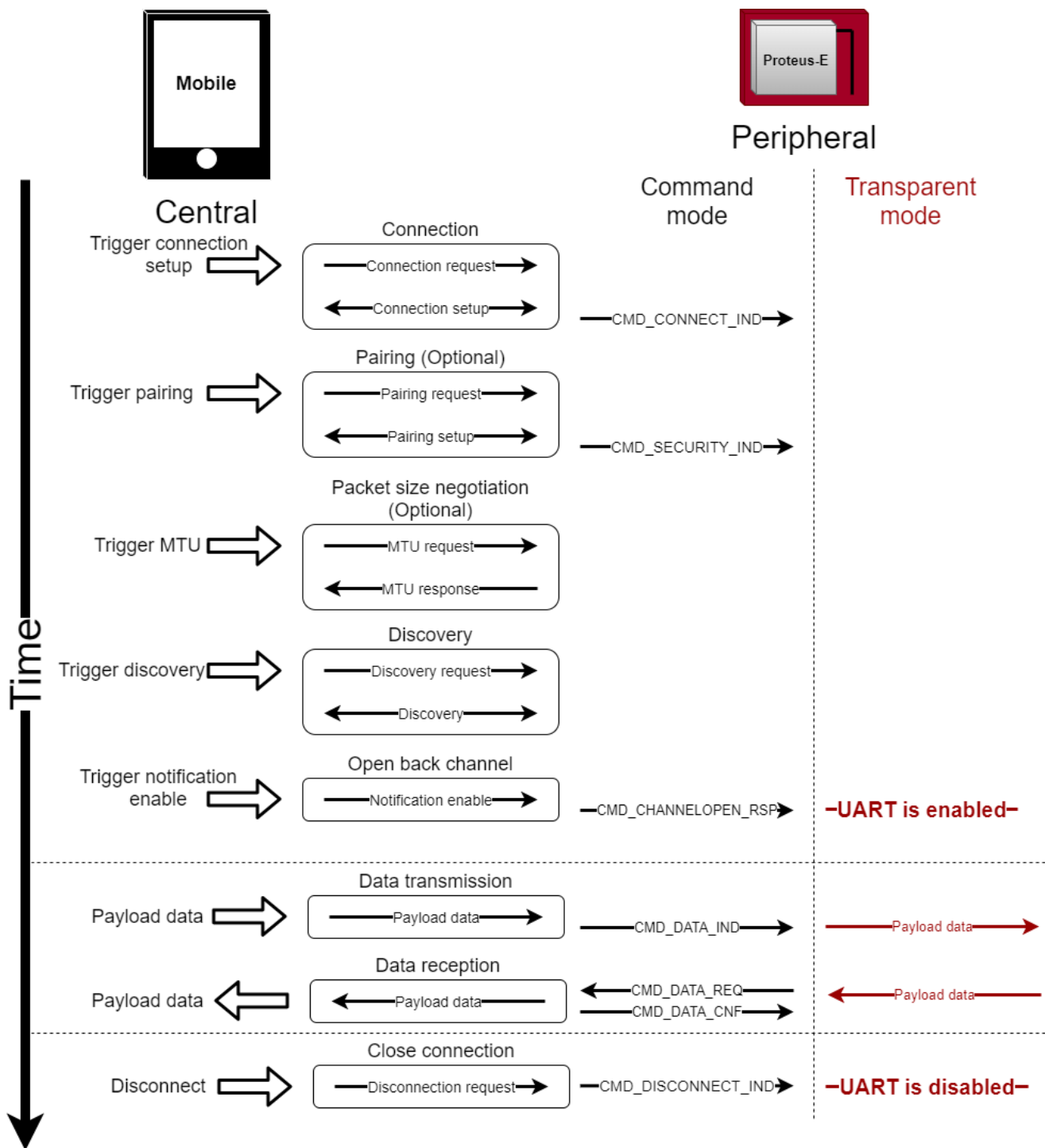


Figure 12: Steps for the connection setup



For more details about the connection setup and data transmission, please refer to application note ANR025 [3].

7.7. Advertising of custom data

The content of standard advertising and scan response packets of the Proteus-e is automatically defined. It contains the device name, the UUID of the Bluetooth® LE profile and the TX-power of the Proteus-e. In few cases, like the transmission of Bluetooth® beacons, the standard content of the advertising and scan response packet is not suitable any-more.



Bluetooth® beacons are advertising and scan response data of a specific format, like Eddystone beacon [4], iBeacon [5] or even Würth Elektronik eiSos beacons [6].

To place custom data (like beacons) in the advertising and scan response packet, the Proteus-e implements the user settings `RF_AdvertisingData` and `RF_ScanResponseData`. Both settings contain the raw data that is to be placed in the advertising packet and scan response packet respectively, 31 bytes maximum per packet.

If these user settings contain data, the standard content is not used any-more in the advertising and scan response packet during advertising.



The format of the raw data is defined in the Bluetooth® specification [1] chapter "ADVERTISING AND SCAN RESPONSE DATA FORMAT". Placing other content in the advertising and scan response packets can result in malfunctioning.

7.7.1. Restrictions

In case custom data shall be placed in the advertising and scan response packet, the following restrictions must be respected:

1. It is not allowed to place "Flags" (header byte 0x01) in the `RF_ScanResponseData`.
2. In case the device name (header byte 0x08 or 0x09) shall be added, it must be placed either in `RF_AdvertisingData` or `RF_ScanResponseData`, not in both at the same time.

If one or several of these restrictions are not respected, custom data can not be written to the user settings `RF_AdvertisingData` and `RF_ScanResponseData`.

7.7.2. Application of custom advertising and scan response data

The content of the standard advertising and scan response packet is only cleared, if the user setting `RF_AdvertisingData` contains at least one byte. In this case the content of `RF_AdvertisingData` is placed in the advertising packet.

If the user setting `RF_ScanResponseData` contains at least one byte in addition, this content is placed in the scan response packet.

Thus, to place custom data in the scan response packet, custom data must be placed in the advertising packet first.

There are two options to place custom data in the settings `RF_AdvertisingData` and `RF_ScanResponseData`:

1. Use the command `CMD_SET_REQ` to write the custom data in the setting placed in flash. The new data is only applied after a device reset. As this command uses one flash cycle it is suited to define the default content of the advertising and the scan response packet.
2. Use the command `CMD_SETRAM_REQ` to write the custom data in the local copy of the setting placed in RAM, which is set to default after device reset. This setting is suited to apply new data in the advertising and the scan response packet during runtime.

7.8. Energy-efficient distance estimation solutions

The transmitted scan response packet contains the TX power value used by the Proteus-e. This value in combination with the RSSI value of the received advertising packet can be used to estimate the distance to the module. Using a suitable triangulation algorithm and multiple receivers or transmitters, a position can be approximately determined.



The scan response packet is only received by the scanner, if it performs an active scan.

7.9. Configure the module for low power consumption

Depending on the application environment of the Proteus-e, the goal is to find the optimal trade-off between the module's performance and its power consumption. Therefore, the main settings and operation modes that affect the current consumption are listed below:

- `CMD_SLEEP_REQ`: This command puts the module into `ACTION_SLEEP` mode, where it consumes the lowest current ($<1\mu\text{A}$). In this case, both the UART and the Bluetooth® LE interface are shut down.
- `CMD_UARTDISABLE_REQ`: This command disables the UART interface. It is enabled again as soon as the module is reset/woken or when the module outputs a message e.g. when a connection request has been received or the `UART_ENABLE` pin of the module was used.
- `RF_TXPower`: This setting can be used to configure the output power of the module in radio transmission mode at the cost of reduced radio range.
- `RF_AdvertisingInterval` and `RF_ConnectionInterval`: These parameters define the timing behaviour of the Proteus-e when advertising or during an open connection. The larger these intervals are, the less often data is transmitted via radio and thus power consumption decreases.
- The 2 MBit radio mode transmits data packets faster and thus reduces the power consumption slightly.

7.10. Start the direct test mode (DTM)

The direct test mode (DTM) enables the test functions described in Bluetooth® Specification. The purpose of DTM is to test the operation of the radio at the physical level, such as transmission power and receiver sensitivity, frequency offset and drift, modulation characteristics, packet error rate and inter modulation performance.

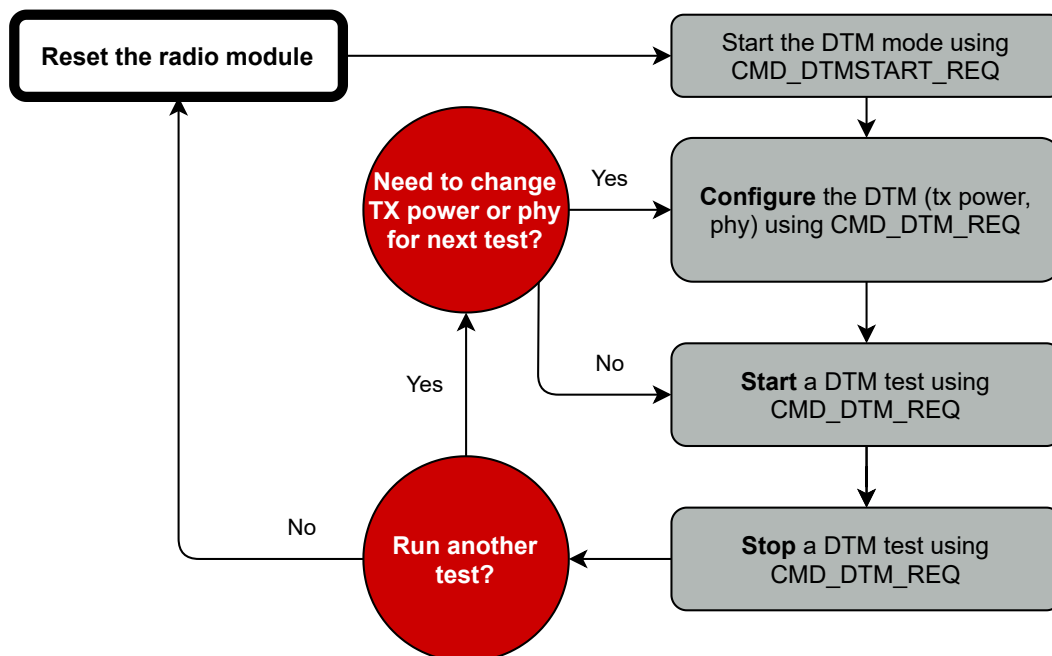


Figure 13: DTM flow chart

Conformance tests of the nRF52 with the DTM application are carried out by dedicated test equipment. To get access to the test functions the `CMD_DTMSTART_REQ`¹ shall be used first. This command restarts the module in direct test mode. A `CMD_GETSTATE_CNF` message confirms that the DTM has been started successfully.

Now the `CMD_DTM_REQ` can be used to configure output power and phy of the subsequent tests. Next, to start and stop the test functions, please use again the `CMD_DTM_REQ` command. After a test has been started, it has to be stopped before a next test can be run.

Example: Transmission test on channel 0 with Bit pattern 0x0F

The goal of this example is to show how the DTM, and in specific the transmission/reception test, can be run. Here fore we need to connect two modules, start the transmission test on one module and start the reception test on the second module. In this section, all packet data from or to the modules is given in **hexadecimal notation**.

All steps are described in the following:

- First, restart the modules in DTM mode.

¹Please note that the WE UART Terminal [7] PC tool in version 1.3.1.0 or newer supports the DTM function.

| Info | Module A | Module B |
|---|----------------------|----------------------|
| ⇒ Request CMD_DTMSTART_REQ to enable the DTM on module A | 02 1D 00 00 1F | |
| ⇐ Response CMD_DTMSTART_CNF: Request understood, try to start DTM now | 02 5D 01 00 00 5E | |
| ⇐ Indication CMD_GETSTATE_CNF: Restarted module with DTM enabled | 02 41 02 00 10 05 54 | |
| ⇒ Request CMD_DTMSTART_REQ to enable the DTM on module B | | 02 1D 00 00 1F |
| ⇐ Response CMD_DTMSTART_CNF: Request understood, try to start DTM now | | 02 5D 01 00 00 5E |
| ⇐ Indication CMD_GETSTATE_CNF: Restarted module with DTM enabled | | 02 41 02 00 10 05 54 |

- Now both modules are ready for the DTM configuration and operation.
- (Optional) Set both modules to 2 MBit mode (default setting after start-up is 1 MBit).

| Info | Module A | Module B |
|--|-------------------------------|-------------------------------|
| ⇒ Request CMD_DTM_REQ to set module A to 2 MBit mode | 02 1E 04 00 00 02 02 00 18 | |
| ⇐ Response CMD_DTM_CNF: Success | 02 5E 03 00 00 00 00 5F | |
| ⇒ Request CMD_DTM_REQ to set module B to 2 MBit mode | | 02 1E 04 00 00 02 02 00 18 |
| ⇐ Response CMD_DTM_CNF: Success | | 02 5E 03 00 00 00 00 5F |

- (Optional) Set the output power to 0 dBm (default setting after start-up is 4 dBm).

| Info | Module A | Module B |
|---|-------------------------------|-------------------------------|
| ⇒ Request CMD_DTM_REQ to set module A to 8 dBm output power | 02 1E 04 00 02 00 02 03 1B | |
| ⇐ Response CMD_DTM_CNF: Success | 02 5E 03 00 00 00 00 5F | |
| ⇒ Request CMD_DTM_REQ to set module B to 8 dBm output power | | 02 1E 04 00 02 00 02 03 1B |
| ⇐ Response CMD_DTM_CNF: Success | | 02 5E 03 00 00 00 00 5F |

- After configuration the tests can start. First start the transmission test.

| Info | Module A | Module B |
|---|-------------------------------|----------|
| ⇒ Request CMD_DTM_REQ to start the transmission test on module A with channel 0 and Bit pattern 16 times 0x0F | 02 1E 04 00 02 00 10 01 0B | |
| ⇐ Response CMD_DTM_CNF: Started test successfully | 02 5E 03 00 00 00 00 5F | |

- Start the reception test.

| Info | Module A | Module B |
|--|----------|-------------------------------|
| ⇒ Request CMD_DTM_REQ to start the reception test on module B with channel 0 | | 02 1E 04 00 01 00 00 00 19 |
| ⇐ Response CMD_DTM_CNF: Started test successfully | | 02 5E 03 00 00 00 00 5F |

- Stop both tests again.

| Info | Module A | Module B |
|---|-------------------------------|-------------------------------|
| ⇒ Request CMD_DTM_REQ to stop the transmission test | 02 1E 04 00 03 00 00 01 1A | |
| ⇐ Response CMD_DTM_CNF: Stopped test successfully | 02 5E 03 00 00 80 00 DF | |
| ⇒ Request CMD_DTM_REQ to stop the reception test | | 02 1E 04 00 03 00 00 01 1A |
| ⇐ Response CMD_DTM_CNF: Stopped test successfully, received 0x14FE (5374 _{dec}) packets | | 02 5E 03 00 00 94 FE 35 |

During the time the reception and transmission tests were running 5374 data packets have been received by module B, which were transmitted by module A.

7.11. Using the 2 MBit phy

Bluetooth® 5 allows to transmit data with 2 MBit data rate.

To be backward compatible to Bluetooth® LE 4.x devices, Bluetooth® LE connections must still be setup using the 1 MBit phy. As soon as a connection has been setup, the connection can be updated to the 2 MBit phy. To switch the phy after the connection has been setup the Proteus-e offers the command CMD_PHYUPDATE_REQ. As response to this request a CMD_PHYUPDATE_IND is returned from the Proteus-e, that gives feedback if the connection was switched to the new phy, or if the connection partner rejected the request.



Please note that the 2 MBit phy is an optional feature of Bluetooth® LE devices since Bluetooth® LE 5.0 which may not be supported by the connection partner.

8. Host connection

8.1. Serial interface: UART

The configuration in factory state of the UART is 115200 Baud without flow control and with data format of 8 data Bits, no parity and 1 stop and 1 start Bit ("8n1"). The baud rate and flow control of the UART can be configured by means of the user setting `UART_ConfigIndex`. The data format is fixed to 8n1.

The output of characters on the serial interface runs with secondary priority. For this reason, short interruptions may occur between the outputs of individual successive bytes. The host must not implement too strict timeouts between two bytes to be able to receive packets that have interruptions in between.

8.1.1. Reset behavior

When holding the module's */RESET* pin LOW, the radio chip states are undefined. In this case the modules *UTXD* pin may be pulled LOW by the radio module, such that the connected host controller's UART may detect a 0x00-byte with frame error.

To guarantee a clean UART communication, the host controller shall discard its UART RX buffer contents and any UART RX error flags, while the Proteus-e's */RESET* pin is held LOW by the host.

9. The command interface

The module acts as a slave and can be fully controlled by an external host. The configuration as well as the operation of the module can be managed by predefined commands that are sent as telegrams over the UART interface of the module.

The commands of the command interface can be divided into 3 groups:

- **Requests:** The host requests the module to trigger an action, e.g. in case of the request `CMD_RESET_REQ` the host asks the module to perform a reset.
- **Confirmations:** On each request, the module answers with a confirmation message to give a feedback on the requested operation status. In case of a `CMD_RESET_REQ`, the module answers with a `CMD_RESET_CNF` to tell the host whether the reset will be performed or not.
- **Indications and responses:** The module indicates spontaneously when a special event has occurred. The `CMD_CONNECT_IND` indicates for example that a connection has been established.

| Start signal | Command | Length | Payload | CS |
|--------------|---------|-------------------|--------------|--------|
| 0x02 | 1 Byte | 2 Byte, LSB first | Length Bytes | 1 Byte |

Start signal: 0x02 (1 Byte)

Command: Command byte identifying the command (1 Byte).

Length: Specifies the length of the payload that follows. Length is a 16 Bit field with LSB first.

Payload: Variable number of data or parameters (defined by the length field).

Checksum (CS): Byte wise XOR combination of all preceding Bytes including the start signal, i.e. $0x02 \oplus \text{Command} \oplus \text{Length} \oplus \text{Payload} = \text{CS}$



Host integration example codes for checksum calculation and command frame structure can be found in annex A and B, as well as in the Wireless Connectivity SDK [8, 9].



If the transmission of the UART command has not finished within the packet transmission duration (depending on the currently selected UART baud rate + 5 ms after having received the start signal), the module discards the received Bytes and waits for a new command. This means that the delay between 2 successive Bytes in a frame must be kept as low as possible.



Please note that the different commands are only valid in specific module states (see Figure 10). If a command is not permitted in the current state, the command confirmation returns "Operation not permitted" as a response. When in sleep state the module will not return any response.

9.1. Setup connections

9.1.1. CMD_CONNECT_IND

This telegram indicates that a remote device has started the connection process. It contains the status and, in case of success, the FS_BTMAC of the connecting device.

Format (connected successfully):

| Start signal | Command | Length | Status | BTMAC | CS |
|--------------|---------|-----------|--------|---------|--------|
| 0x02 | 0x86 | 0x07 0x00 | 0x00 | 6 Bytes | 1 Byte |

Format (failed to connect):

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0x86 | 0x01 0x00 | 0x01 | 1 Byte |

Status:

0x00: Physical connection established successfully

0x01: Connection failed, e.g. due to a timeout

9.1.2. CMD_SECURITY_IND

This telegram indicates the security status and the FS_BTMAC of the connected device.

Format:

| Start signal | Command | Length | Status | BTMAC | CS |
|--------------|---------|-----------|--------|---------|--------|
| 0x02 | 0x88 | 0x07 0x00 | 1 Byte | 6 Bytes | 1 Byte |

Status:

0x00: Encrypted link to previously bonded device established

0x01: Bonding successful, encrypted link established

0x02: No bonding, pairing successful, encrypted link established

9.1.3. CMD_CHANNELOPEN_RSP

This command is sent to the host as soon as connection setup has been completed successfully. Now data can be transmitted using the CMD_DATA_REQ. Next to the FS_BTMAC of the connected device, the maximum payload size (MPS) that is supported by the link is part of this telegram.

Format:

| Start signal | Command | Length | Status | BTMAC | MPS | CS |
|--------------|---------|-----------|--------|---------|--------|--------|
| 0x02 | 0xC6 | 0x08 0x00 | 1 Byte | 6 Bytes | 1 Byte | 1 Byte |

Status:

0x00: Success

9.1.4. CMD_DISCONNECT_REQ

This command closes the existing connection. Thereafter the module prints a CMD_DISCONNECT_CNF to confirm that the request has been received. The indication message CMD_DISCONNECT_IND follows which determines whether the disconnection operation has been performed successfully or not.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x07 | 0x00 0x00 | 0x05 |

Response (CMD_DISCONNECT_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x47 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, try to disconnect

0x01: Operation failed

0xFF: Operation not permitted

9.1.5. CMD_DISCONNECT_IND

This telegram indicates that the connection has shut down successfully. This indication message is the result of a disconnection request (CMD_DISCONNECT_REQ).

Format:

| Start signal | Command | Length | Reason | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0x87 | 0x01 0x00 | 1 Byte | 1 Byte |

Reason:

0x08: Connection timeout

0x13: User terminated connection

0x16: Host terminated connection

0x3B: Connection interval unacceptable

0x3D: Connection terminated due to MIC failure (Not able to connect due to bad link quality, or connection request ignored due to wrong key)

0x3E: Connection setup failed

9.1.6. CMD_PHYUPDATE_REQ

This command allows to update the PHY of the current Bluetooth® LE connection. After the module prints a CMD_PHYUPDATE_CNF it tries to update the PHY. The result is indicated by CMD_PHYUPDATE_IND message.

Format:

| Start signal | Command | Length | PHY | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0x1A | 0x01 0x00 | 1 Byte | 1 Byte |

PHY:

0x01: 1 MBit PHY

0x02: 2 MBit PHY

Response (CMD_PHYUPDATE_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x5A | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received. Try to update PHY of current connection

0x01: Operation failed, e.g. due to invalid PHY

0xFF: Operation not permitted

9.1.7. CMD_PHYUPDATE_IND

This command indicates that there was an attempt to update the PHY of the existing connection. If the PHY update was successful, the command includes the new PHY for receiving and transmitting direction, as well as the BTMAC of the device connected to. This command is the result of the CMD_PHYUPDATE_REQ.

Format in case of success:

| Start signal | Command | Length | Status | PHY Rx | PHY Tx | BTMAC | CS |
|--------------|---------|-----------|--------|--------|--------|---------|--------|
| 0x02 | 0x9A | 0x09 0x00 | 0x00 | 1 Byte | 1 Byte | 6 Bytes | 1 Byte |

PHY Rx/PHY Tx:

0x01: Using 1 MBit PHY now

0x02: Using 2 MBit PHY now

Format in case of failure:

| Start signal | Command | Length | Status | Info | CS |
|--------------|---------|-----------|--------|--------|--------|
| 0x02 | 0x9A | 0x02 0x00 | 0x01 | 1 Byte | 1 Byte |

Info:

0x1A: Unsupported feature of remote device

9.1.8. CMD_GETBONDS_REQ

This command requests the MAC addresses of all bonded devices.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x0F | 0x00 0x00 | 0x0D |

Response (CMD_GETBONDS_CNF):

| Start signal | Command 0x40 | Length | Status | #Devices | Payload | CS |
|--------------|----------------|---------|--------|----------|--------------------|--------|
| 0x02 | 0x4F | 2 Bytes | 1 Byte | 1 Byte | (Length - 2) Bytes | 1 Byte |

The Payload sequentially lists the data of the bonded #Devices devices. It consists of #Devices times the following telegram (see example below).

| Bond_ID | BTMAC |
|---------|---------|
| 2 Bytes | 6 Bytes |

Status:

0x00: Request successfully processed

0x01: Operation failed

0xFF: Operation not permitted

9.1.8.1. Example 1

Request for the bonding data of the devices in database.

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x0F | 0x00 0x00 | 0x0D |

Response:

| Start signal | Command 0x40 | Length | Status | #Devices | Payload | CS |
|--------------|-------------------|-----------|--------|----------|--|------|
| 0x02 | 0x4F | 0x12 0x00 | 0x00 | 0x02 | 0x00 0x00 0x82 0x5C 0xA7 0xE2 0x87 0xD0 0x01 0x00 0x01 0x00 0x00 0xDA 0x18 0x00 | 0x53 |

Two devices have been bonded before:

- Device 1 (Bond_ID 0x0000) with FS_BTMAC 0x82 0x5C 0xA7 0xE2 0x87 0xD0
- Device 2 (Bond_ID 0x0001) with FS_BTMAC 0x01 0x00 0x00 0xDA 0x18 0x00

9.1.9. CMD_DELETEBONDS_REQ

This command removes the bonding information of all or single bonded devices. Enter Bond_ID to remove the bonding data of a certain Bond_ID. To remove all bonding data, choose Length equals 0 and leave Bond_ID empty.

Format:

| Start signal | Command | Length | Bond_ID | CS |
|--------------|---------|---------|--------------|--------|
| 0x02 | 0x0E | 2 Bytes | 0 or 2 Bytes | 1 Byte |

Response (CMD_DELETEBONDS_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x4E | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request successfully processed

0x01: Operation failed (e.g. Bond_ID not found)

0xFF: Operation not permitted

9.1.9.1. Example 1

Request to remove all bonding data.

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x0E | 0x00 0x00 | 0x0C |

Response:

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x4E | 0x01 0x00 | 0x00 | 0x4D |

Successfully removed all bonding information.

9.1.9.2. Example 2

Request to remove the bonding of the device corresponding to Bond_ID 0.

| Start signal | Command | Length | Bond_ID | CS |
|--------------|---------|-----------|-----------|------|
| 0x02 | 0x0E | 0x02 0x00 | 0x00 0x00 | 0x0E |

Response:

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x4E | 0x01 0x00 | 0x00 | 0x4D |

Successfully removed the bonding information.

9.1.10. CMD_ALLOWUNBONDEDCONNECTIONS_REQ

In case the SECFLAGS_BONDEDCONNECTIONSONLY_ENABLE bit has been set in the RF_SecFlags user setting, this command temporarily allows the connection setup of unbonded devices until the radio module is reset.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x2D | 0x00 0x00 | 0x2F |

Response (CMD_ALLOWUNBONDEDCONNECTIONS_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|---------|--------|--------|
| 0x02 | 0x6D | 2 Bytes | 1 Byte | 1 Byte |

Status:

0x00: Request successfully processed

0x01: Operation failed

0xFF: Operation not permitted

9.2. Transmit and receive data

9.2.1. CMD_DATA_REQ

This command provides the simple data transfer to the previously connected device. This command is suitable for transmission for a point-to-point connection. The number of payload data bytes (MPS) is negotiated during the connection phase. It can be maximal 243 bytes, but at least 19 bytes.

When the data is processed by the module a CMD_DATA_CNF is sent to the host. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been sent.

The receiving Proteus-e will get a CMD_DATA_IND message containing the transmitted payload data.

Format:

| Start signal | Command | Length | Payload | CS |
|--------------|---------|---------|--------------|--------|
| 0x02 | 0x04 | 2 Bytes | Length Bytes | 1 Byte |

Response (CMD_DATA_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|---------|--------------|--------|
| 0x02 | 0x44 | 2 Bytes | Length Bytes | 1 Byte |

Status:

0x00: Request received, will send data now

0x01 + 0xXX: Operation failed + 0xXX maximum payload size (if it was exceeded)

0xFF: Operation not permitted

9.2.2. CMD_TXCOMPLETE_RSP

This command is sent to the host as soon as the data, which was requested by a CMD_DATA_REQ has been transmitted.

Format:

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0xC4 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Data transmitted successfully

0x01: Data transmission failed

9.2.3. CMD_DATA_IND

This telegram indicates the reception of data sent by the previously connected device. The CMD_DATA_IND returns the FS_BTMAC of the sending device, the RSSI value of the received data packet and the payload data received. The RSSI value is printed in two's complement notation.

Format:

| Start signal | Command | Length | BTMAC | RSSI | Payload | CS |
|--------------|---------|---------|---------|--------|--------------------|--------|
| 0x02 | 0x84 | 2 Bytes | 6 Bytes | 1 Byte | (Length - 7) Bytes | 1 Byte |

9.3. Configuring the module and modifying the device settings

The module's parameters are stored in flash, but have a local copy in RAM. The flash parameters can be modified by the `CMD_SET_REQ`, read by the `CMD_GET_REQ` and retain their content even when resetting the module.

A few chosen settings can also be updated during runtime in the local RAM copy using `CMD_SETRAM_REQ` and `CMD_GETRAM_REQ`, that take effect immediately without the need of a device reset.

9.3.1. `CMD_SET_REQ`

This command enables direct manipulation of the parameters in the module's settings in flash. The respective parameters are accessed by means of the corresponding settings index, which can be found in Table 22.

Parameters of two or more Bytes have to be transferred with the LSB first unless noted differently in the corresponding description.



The modified parameters only take effect after a restart of the module. This may be done by a `CMD_RESET_REQ` if the module does not restart automatically.



The flash memory used to store these settings has a limited count of at least 10,000 write cycles. Try to avoid performing periodic `CMD_SET_REQ` as each command will use one write cycle.



The validity of the specified parameters is not verified. Incorrect values can result in device malfunction!



To save the parameters in the flash memory of the module, the particular memory segment must first be flushed entirely and then restored from RAM. If a reset occurs during this procedure, the entire memory area may be corrupted (e.g. due to supply voltage fluctuations).

Recommendation: First, verify the configuration of the module with `CMD_GET_REQ` and only then apply a `CMD_SET_REQ` if required to avoid unnecessary flash cycles.

Format:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|---------|----------------|--------------------|--------|
| 0x02 | 0x11 | 2 Bytes | 1 Byte | (Length - 1) Bytes | 1 Byte |

Response (`CMD_SET_CNF`):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x51 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, settings set successfully

0x01: Operation failed due to invalid parameter

0x04: Serious error, when writing flash. Try to factory reset or re-flash the device

0x05: Supply voltage is below 1.8 V $\pm 5\%$. Please apply correct supply voltage, reset and retry.

0xFF: Operation not permitted

9.3.1.1. Example 1

Setting the advertising time RF_AdvertisingTimeout to 180 seconds.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x07 | 0xB4 0x00 | 0xA3 |

Response:

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

Setting was set successfully.

9.3.1.2. Example 2

Setting the static pass key RF_StaticPasskey to "123456".

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-------------------------------|------|
| 0x02 | 0x11 | 0x07 0x00 | 0x12 | 0x31 0x32 0x33 0x34 0x35 0x36 | 0x01 |

Response:

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

Setting was set successfully.

9.3.2. CMD_GET_REQ

This command can be used to query individual setting parameters in flash. The respective parameters are accessed by means of the corresponding settings index, which can be found in Table 22.

Parameters of two or more bytes have to be transferred with the LSB first unless noted differently in the corresponding description.

Read access to the memory area outside the setting is blocked.

Format:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|--------|
| 0x02 | 0x10 | 0x01 0x00 | 1 Byte | 1 Byte |

Response (CMD_GET_CNF):

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|---------|--------|--------------------|--------|
| 0x02 | 0x50 | 2 Bytes | 1 Byte | (Length - 1) Bytes | 1 Byte |

Status:

0x00: Request received, read out of setting successful

0x01: Operation failed

0xFF: Operation not permitted

9.3.2.1. Example 1

Request the current static pass key RF_StaticPasskey.

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x12 | 0x01 |

Response: The current RF_StaticPasskey in flash is "123123" (0x31 0x32 0x33 0x31 0x32 0x33).

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-------------------------------|------|
| 0x02 | 0x50 | 0x07 0x00 | 0x00 | 0x31 0x32 0x33 0x31 0x32 0x33 | 0x55 |

Setting was read successfully.

9.3.3. CMD_SETRAM_REQ

This command enables direct manipulation of the parameters in the module's settings in the local RAM copy. Changes to the parameters are applied immediately. The respective parameters are accessed by means of the corresponding settings index, which can be found in Table 23. Parameters of two or more Bytes have to be transferred with the LSB first unless noted differently in the corresponding description.



The validity of the specified parameters is not verified. Incorrect values can result in device malfunction!

Format:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|---------|----------------|--------------------|--------|
| 0x02 | 0x21 | 2 Bytes | 1 Byte | (Length - 1) Bytes | 1 Byte |

Response (CMD_SETRAM_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x61 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, settings set successfully

0x01: Operation failed due to invalid parameter

0x04: Serious error

0xFF: Operation not permitted

9.3.3.1. Example 1

Setting the scan response packet `RF_ScanResponseData` to "TX power 4dBm" (0x02 0x0A 0x04) seconds.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|----------------|------|
| 0x02 | 0x21 | 0x04 0x00 | 0x0E | 0x02 0x0A 0x04 | 0x25 |

Response:

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x61 | 0x01 0x00 | 0x00 | 0x52 |

Setting was set successfully.

9.3.4. CMD_GETRAM_REQ

This command can be used to query individual setting parameters in RAM. The respective parameters are accessed by means of the corresponding settings index, which can be found in Table 23.

Parameters of two or more bytes have to be transferred with the LSB first unless noted differently in the corresponding description.

Read access to the memory area outside the setting is blocked.

Format:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|--------|
| 0x02 | 0x20 | 0x01 0x00 | 1 Byte | 1 Byte |

Response (CMD_GET_CNF):

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|---------|--------|--------------------|--------|
| 0x02 | 0x60 | 2 Bytes | 1 Byte | (Length - 1) Bytes | 1 Byte |

Status:

0x00: Request received, read out of setting successful

0x01: Operation failed

0xFF: Operation not permitted

9.3.4.1. Example 1

Request the current scan response packet `RF_ScanResponseData`.

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x20 | 0x01 0x00 | 0x0E | 0x2D |

Response: The current `RF_ScanResponseData` in RAM is "TX power 4dBm" (0x02 0x0A 0x04).

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------|------|
| 0x02 | 0x60 | 0x04 0x00 | 0x00 | 0x02 0x0A 0x04 | 0x6A |

Setting was read successfully.

9.4. Manage the device state

9.4.1. CMD_GETSTATE_REQ

This command returns the current state of the module.



Please refer to chapter 7 for details on the states of the module.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x01 | 0x00 0x00 | 0x03 |

Response (CMD_GETSTATE_CNF):

| Start signal | Command 0x40 | Length | Module role | Module actions | More info | CS |
|--------------|-------------------|---------|-------------|-------------------|--------------------|--------|
| 0x02 | 0x41 | 2 Bytes | 1 Byte | 1 Byte | (Length - 2) Bytes | 1 Byte |

Module role:

0x00: No role

0x01: Peripheral

0x10: Direct test mode (DTM)

Other: Reserved

Module action:

0x00: No action

0x01: Idle (advertising)

0x03: Connected (More info is the 6 Bytes FS_BTMAC address of the connected device, followed by the MPS of the current connection)

0x04: Sleep (system-off mode)

0x05: Direct test mode (More info is 1 reserved byte)

0x06: Connected, but link is still closed (central must enable notifications first)

9.4.1.1. Example 1

Get the current state of the module.

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x01 | 0x00 0x00 | 0x03 |

Response:

| Start signal | Command 0x40 | Length | Module role | Module actions | More info | CS |
|--------------|-------------------|-----------|----------------|-------------------|---|------|
| 0x02 | 0x41 | 0x09 0x00 | 0x01 | 0x03 | 0x11 0x00 0x00 0xDA 0x18 0x00 0xF3 | 0x68 |

The module is connected to another module with FS_BTMAC 0x11 0x00 0x00 0xDA 0x18 0x00 and an MPS of 243 Bytes (0xF3).

9.4.2. CMD_RESET_REQ

This command triggers a software reset of the module.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x00 | 0x00 0x00 | 0x02 |

Response (CMD_RESET_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x40 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, will perform reset now

0x01: Operation failed

0xFF: Operation not permitted

9.4.3. CMD_SLEEP_REQ

This command is used to start the system-off mode (ACTION_SLEEP). For more details, see chapter 7.4.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x02 | 0x00 0x00 | 0x00 |

Response (CMD_SLEEP_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x42 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, will go to sleep now

0x01: Operation failed

0xFF: Operation not permitted

9.4.4. CMD_SLEEP_IND

This indication is sent by the module when the `RF_AdvertisingTimeout` has expired without a connection to the module.

Format:

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0x82 | 0x01 0x00 | 0x00 | 1 Byte |

Status:

0x00: Advertising timeout detected, will go to sleep now

9.4.5. CMD_FACTORYRESET_REQ

This command triggers a factory reset of the module. First, the default user settings are re-stored, then the module is reset.



This command also removes all bonding data and GPIO configurations.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x1C | 0x00 0x00 | 0x1E |

Response (CMD_FACTORYRESET_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x5C | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, will perform factory reset now

0x01: Operation failed

0xFF: Operation not permitted



To save the parameters in the flash memory of the module, the particular memory segment must first be flushed entirely and then restored from RAM. If a reset occurs during this procedure (e.g. due to supply voltage fluctuations), the entire memory area may be destroyed.



During start-up of the device, the user settings memory is checked for consistency. In case of inconsistency (e.g. the memory was erased) the device will perform a factory reset.

9.4.6. CMD_UARTDISABLE_REQ

This command disables the UART of the module.

It will be re-enabled when the module has to send data to the host (e.g. data was received via radio or a state is indicated) or if the *UART_ENABLE* pin is used (apply a falling edge, hold low for at least 15 ms before applying a rising edge and hold high for at least 15 ms). In this case, either the received data or a *CMD_UARTENABLE_IND* is transmitted by the module. Afterwards the UART will stay active until another *CMD_UARTDISABLE_REQ* is sent to the Proteus-e.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x1B | 0x00 0x00 | 0x19 |

Response (*CMD_UARTDISABLE_CNF*):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x5B | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, will disable UART now

0x01: Operation failed

0xFF: Operation not permitted



It is strongly recommended to disable the UART only, if it is foreseeable that there will be no UART communication for several seconds. Use cases could be during advertising phase to wait for connecting Bluetooth® LE devices.



Disabling the UART peripheral of the module results in a reduction of current consumption of about 250 μ A.

9.4.7. CMD_UARTENABLE_IND

This indication is shown when the UART of the module is re-enabled (after performing a CMD_UARTDISABLE_REQ followed by using the *UART_ENABLE* pin for wake-up). After receiving this message the UART can be used for any operation again.

Format:

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0x9B | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: UART has been re-enabled successfully

9.5. Run the Bluetooth test modes

The test modes "DTM" as specified by the Bluetooth® SIG are defined in the Bluetooth® Core specification.

9.5.1. CMD_DTMSTART_REQ

This command restarts the module in direct test mode (DTM). When starting in DTM mode, a CMD_GETSTATE_CNF message follows which indicates that the test mode has been enabled successfully. Now the CMD_DTM_REQ can be used to start and stop various test modes. Performing a reset will leave the DTM and restart the module in the ACTION_IDLE state.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x1D | 0x00 0x00 | 0x1F |

Response (CMD_DTMSTART_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x5D | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x00: Request received, will enable the direct test mode now

0x01: Operation failed

0xFF: Operation not permitted

9.5.2. CMD_DTM_REQ

This command starts and stops various test modes. To be able to run these test modes, the DTM has to be enabled first using the CMD_DTMSTART_REQ. After a test has been started, it has to be stopped first before a next test can be run.

The default TX power value is 8 dBm, the allowed range is from -40 up to +4 dBm (see chapter 10.14 for valid TX power values). The valid range for channel is 0...39.



Configuring the phy and TX power can be only done, when the test is stopped (no TX or RX test ongoing).

Format:

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|--------|
| 0x02 | 0x1E | 0x04 0x00 | 1 Byte | 1 Byte | 1 Byte | 1 Byte | 1 Byte |

Command code:

0x00: DTM setup

| Vendor option | Vendor command | Payload |
|-----------------|---|---------|
| 0x00: Reset DTM | 0x00 | 0x00 |
| 0x02: Set phy | New phy 1. 0x01: 1MBit 2. 0x02: 2MBit | 0x00 |

0x01: Start RX test

| Channel | Length | Payload |
|--|--------|---------|
| Frequency = (2402 + Channel * 2) MHz to be used for RX | 0x00 | 0x00 |

0x02: Start TX test

| Channel | Length | Payload |
|--|------------------------------|---|
| Frequency = (2402 + Channel * 2) MHz to be used for TX | Length of the packet to send | Bit pattern 0x00: PRBS9 0x01: 0x0F 0x02: 0x55 |

| Vendor option | Vendor command | Payload |
|---|--------------------|------------------------------|
| Frequency = (2402 + Channel * 2) MHz to be used for TX | 0x00: Carrier test | 0x03: Vendor specific |
| TX power -40 up to +4 dBm (see chapter 10.14 for valid TX power values) | 0x02: Set TX power | 0x03: Vendor specific |

0x03: Stop last test

| Channel | Length | Payload |
|---------|--------|---------|
| 0x00 | 0x00 | 0x00 |

Response (CMD_DTM_CNF):

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|---------|--------|-----------|--------|
| 0x02 | 0x5E | 2 Bytes | 1 Byte | 0-2 Bytes | 1 Byte |

Status:

0x00: Request received

0x01: Operation failed

0x03: Busy

0xFF: Operation not permitted

Result:

0x0000: Test success

0x0001: Test failed

0x8000 + n: Received n packets during RX test



See also the example in chapter 7.10.

9.5.2.1. Example: Transmission, 16 times 0x0F, channel 0

Start the transmission test on channel 0 (2402 MHz). The packets consist of 16 times 0x0F:

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x02 | 0x00 | 0x10 | 0x01 | 0x0B |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x5F |

Test started successfully. Now stop the test again.

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x03 | 0x00 | 0x00 | 0x01 | 0x1A |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x80 0x00 | 0xDF |

Test stopped successfully and received 0 packets.

9.5.2.2. Example: Receiver, channel 0

Start the reception test on channel 0 (2402 MHz):

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x01 | 0x00 | 0x00 | 0x00 | 0x19 |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x5F |

Test started successfully. In between we started the transmission test on a second module. When we stop RX test now, we can count the received packets from the transmitting module.

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x03 | 0x00 | 0x00 | 0x01 | 0x0B |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x8E 0x67 | 0xB6 |

Test stopped successfully and received 0x0E67 (3687) packets.

9.5.2.3. Example: Transmission, carrier test, channel 0

Start the carrier test on channel 0 (2402 MHz). We need to use a vendor specific command:

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x02 | 0x00 | 0x00 | 0x03 | 0x19 |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x5F |

See the previous example to stop the test again.

9.5.2.4. Example: Set TX power to -4 dBm

Set the TX power to -4dBm (0xFC in two's complement notation):

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x02 | 0xFC | 0x02 | 0x03 | 0xE7 |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x5F |

9.5.2.5. Example: Set PHY to 2MBit mode

Set the phy to 2MBit mode:

| Start signal | Command | Length | Command code | Channel / Vendor option | Length / Vendor command | Payload | CS |
|--------------|---------|--------------|--------------|-------------------------|-------------------------|---------|------|
| 0x02 | 0x1E | 0x04 0x00 | 0x00 | 0x02 | 0x02 | 0x00 | 0x18 |

Response:

| Start signal | Command 0x40 | Length | Status | Result | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x5E | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x5F |

9.6. Switching GPIOs by remote control

This chapter contains the commands to use the GPIO feature of the Proteus-e. Please refer to chapter 13 for a detailed description.

9.6.1. CMD_GPIO_LOCAL_WRITECONFIG_REQ

This command configures the free GPIOs of the radio module. This is necessary to allow local and remote GPIO control. As the configuration is stored in flash, it is retained after restarting the device.



The flash memory used to store these settings has a limited count of at least 10,000 write cycles. Try to avoid performing periodic CMD_GPIO_LOCAL_WRITECONFIG_REQ as each command will use one write cycle.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x25 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_LOCAL_WRITECONFIG_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x65 | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted

CMD_GPIO_LOCAL_WRITECONFIG_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID | Function | Values |
|--------|---------|----------|-------------------|
| 1 Byte | 1 Byte | 1 Byte | (Length - 2) Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Function:

0x00: GPIO disconnected

0x01: GPIO works as input

0x02: GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x03:
0x00: value field must use 0x00.
- if **Function** is input, Length is 0x03:
0x00: GPIO has no pull resistor
0x01: GPIO has internal pull down resistor
0x02: GPIO has internal pull up resistor
- if **Function** is output, Length is 0x03:
0x00: GPIO is output LOW
0x01: GPIO is output HIGH

CMD_GPIO_LOCAL_WRITECONFIG_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Status |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Status:

0x00: Success

0x01: Failed

9.6.1.1. Example: Configure two GPIOs to output high

Configure the GPIOs with ID **0x01** and **0x02** to output high:

| Start signal | Command | Length | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|----------------------------|----------------------------|------|
| 0x02 | 0x25 | 0x08 0x00 | 0x03 0x01 0x02 0x01 | 0x03 0x02 0x02 0x01 | 0x2C |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|--------|-----------------------|-----------------------|------|
| 0x02 | 0x65 | 0x07 0x00 | 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x00 | 0x63 |

Configured both GPIOs with success.

9.6.2. CMD_GPIO_LOCAL_READCONFIG_REQ

This command reads the current configuration of the free GPIOs of the radio module.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x2B | 0x00 0x00 | 0x29 |

Response (CMD_GPIO_LOCAL_READCONFIG_CNF):

| Start signal | Command | 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|---------|------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x6B | | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted

CMD_GPIO_LOCAL_READCONFIG_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Function | Values |
|--------|---------|----------|-------------------|
| 1 Byte | 1 Byte | 1 Byte | (Length - 2) Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Function:

0x00: GPIO is disconnected

0x01: GPIO works as input

0x02: GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x02:
Values field is not used in this Block
- if **Function** is input, Length is 0x03:
0x00: GPIO has no pull resistor
0x01: GPIO has pull down resistor
0x02: GPIO has pull up resistor

- if **Function** is output, Length is 0x03:
0x00: GPIO is output LOW
0x01: GPIO is output HIGH

9.6.2.1. Example: Read the current GPIO configuration

Read the current configuration:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x2B | 0x00 0x00 | 0x29 |

Response:

| Start signal | Command 0x40 | Length | Status | Blocks | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x6B | 0x09 0x00 | 0x00 | 0x03 0x01 0x02 0x01 0x03 0x02 0x02 0x01 | 0x63 |

The GPIOs with GPIO_ID **0x01** and **0x02** are output high.

9.6.3. CMD_GPIO_REMOTE_WRITECONFIG_REQ

This command configures the free GPIOs of the connected remote device. This is necessary to allow remote GPIO control. As the configuration is stored in flash, it is retained after restarting the device. This command can be run successfully only if the remote device is connected via Bluetooth® LE.



The flash memory used to store these settings has a limited count of at least 10,000 write cycles. Try to avoid performing periodic CMD_GPIO_REMOTE_WRITECONFIG_REQ as each command will use one write cycle.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x28 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_WRITECONFIG_CNF):

| Start signal | Command | 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|---------|------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x68 | | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted

CMD_GPIO_REMOTE_WRITECONFIG_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID | Function | Values |
|--------|---------|----------|-------------------|
| 1 Byte | 1 Byte | 1 Byte | (Length - 2) Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Function:

0x00: GPIO disconnected

0x01: GPIO works as input

0x02: GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x03:
0x00: value field must use 0x00.
- if **Function** is input, Length is 0x03:
0x00: GPIO has no pull resistor
0x01: GPIO has internal pull down resistor
0x02: GPIO has internal pull up resistor
- if **Function** is output, Length is 0x03:
0x00: GPIO is output LOW
0x01: GPIO is output HIGH

CMD_GPIO_REMOTE_WRITECONFIG_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Status |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Status:

0x00: Success

0x01: Failed

0xFF: Remote configuration not allowed (blocked by the user setting CFG_Flags of the remote device)

9.6.3.1. Example: Configure two GPIOs of the connected remote device to output high

Configure the GPIOs with ID **0x01** and **0x02** to output high:

| Start signal | Command | Length | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|----------------------------|----------------------------|------|
| 0x02 | 0x28 | 0x08 0x00 | 0x03 0x01 0x02 0x01 | 0x03 0x02 0x02 0x01 | 0x21 |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|--------|-----------------------|-----------------------|------|
| 0x02 | 0x68 | 0x07 0x00 | 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x00 | 0x6E |

Configured both GPIOs with success.

9.6.4. CMD_GPIO_REMOTE_READCONFIG_REQ

This command reads the current configuration of the free GPIOs of the connected remote device.

Format:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x2C | 0x00 0x00 | 0x2E |

Response (CMD_GPIO_REMOTE_READCONFIG_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x6C | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted

CMD_GPIO_REMOTE_READCONFIG_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Function | Values |
|--------|---------|----------|-------------------|
| 1 Byte | 1 Byte | 1 Byte | (Length - 2) Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Function:

0x00: GPIO is disconnected

0x01: GPIO works as input

0x02: GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x02:
Values field is not used in this Block
- if **Function** is input, Length is 0x03:
0x00: GPIO has no pull resistor
0x01: GPIO has pull down resistor

- 0x02:** GPIO has pull up resistor
- if **Function** is output, Length is 0x03:
- 0x00:** GPIO is output LOW
- 0x01:** GPIO is output HIGH

9.6.4.1. Example: Read the current GPIO configuration of the connected remote device

Read the current GPIO configuration of the connected remote device:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x2C | 0x00 0x00 | 0x2E |

Response:

| Start signal | Command 0x40 | Length | Status | Blocks | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x6C | 0x09 0x00 | 0x00 | 0x03 0x01 0x02 0x01 0x03 0x02 0x02 0x01 | 0x64 |

The GPIOs with GPIO_ID **0x01** and **0x02** are output high.

9.6.5. CMD_GPIO_REMOTE_WRITE_REQ

This command writes the free GPIOs of the remote device. This command can be only run successfully if the respective pins of the remote device have been configured as output pins before and the remote device is connected via Bluetooth® LE.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x29 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_WRITE_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x69 | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted (i.e. no device connected via Bluetooth® LE)

CMD_GPIO_REMOTE_WRITE_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID | Value |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Value:

- if **Function** is output

0x00: Set GPIO to LOW

0x01: Set GPIO to HIGH

CMD_GPIO_REMOTE_WRITE_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Status |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Status:

0x00: Success

0x01: Failed

9.6.5.1. Example: Set a remote output GPIO to low

Set the output GPIO (GPIO_ID **0x01**) of the connected remote device to low:

| Start signal | Command | Length | Block ₁ | CS |
|--------------|---------|-----------|-----------------------|------|
| 0x02 | 0x29 | 0x03 0x00 | 0x02 0x01 0x00 | 0x2B |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | CS |
|--------------|----------------|-----------|--------|-----------------------|------|
| 0x02 | 0x69 | 0x04 0x00 | 0x00 | 0x02 0x01 0x00 | 0x6C |

Successfully set GPIO with GPIO_ID **0x01** to low.

9.6.6. CMD_GPIO_REMOTE_READ_REQ

This command reads the free GPIOs of the remote device. This command can be only run successfully if the respective pins of the remote device have been configured as output or input pins before and the remote device is connected via Bluetooth® LE.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x2A | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_READ_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x6A | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted (i.e. no device connected via Bluetooth® LE)

CMD_GPIO_REMOTE_READ_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID ₁ | ... | GPIO_ID _n |
|---------|----------------------|-----|----------------------|
| 1 Bytes | 1 Byte | | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

CMD_GPIO_REMOTE_READ_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Value |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Value:

- if **Function** is output or input
0x00: The remote GPIO is LOW.
0x01: The remote GPIO is HIGH.
0xFF: Failed reading remote GPIO value.

9.6.6.1. Example: Read the values of remote GPIOs

Read the value of the GPIOs with GPIO_ID **0x01** and **0x02** of the connected remote device:

| Start signal | Command | Length | Block ₁ | CS |
|--------------|---------|-----------|-----------------------|------|
| 0x02 | 0x2A | 0x03 0x00 | 0x02 0x01 0x02 | 0x2A |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|--------|-----------------------|-----------------------|------|
| 0x02 | 0x6A | 0x07 0x00 | 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0x6D |

Successfully read the values of the remote GPIOs with GPIO_ID **0x01** (GPIO is low) and **0x02** (GPIO is high).

9.6.7. CMD_GPIO_LOCAL_WRITE_REQ

This command writes the free GPIOs of the local device. This command can be only run successfully if the respective pins of the local device have been configured as output pins before.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x26 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_LOCAL_WRITE_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x66 | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted (i.e. no device connected via Bluetooth® LE)

CMD_GPIO_LOCAL_WRITE_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID | Value |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Value:

- if **Function** is output

0x00: Set GPIO to LOW

0x01: Set GPIO to HIGH

CMD_GPIO_LOCAL_WRITE_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Status |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Status:

0x00: Success

0x01: Failed

9.6.7.1. Example: Set a local output GPIO to low

Set the output GPIO (GPIO_ID **0x01**) of the local device to low:

| Start signal | Command | Length | Block ₁ | CS |
|--------------|---------|-----------|-----------------------|------|
| 0x02 | 0x26 | 0x03 0x00 | 0x02 0x01 0x00 | 0x24 |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | CS |
|--------------|----------------|-----------|--------|-----------------------|------|
| 0x02 | 0x66 | 0x04 0x00 | 0x00 | 0x02 0x01 0x00 | 0x63 |

Successfully set GPIO with GPIO_ID **0x01** to low.

9.6.8. CMD_GPIO_LOCAL_READ_REQ

This command reads the free GPIOs of the local device. This command can be only run successfully if the respective pins of the local device have been configured as output or input pins before.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0x27 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_LOCAL_READ_CNF):

| Start signal | Command 0x40 | Length | Status | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0x67 | 2 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

Status:

0x00: Request received and processed

0x01: Operation failed

0xFF: Operation not permitted (i.e. no device connected via Bluetooth® LE)

CMD_GPIO_LOCAL_READ_REQ block structure

Each **Block** has the following format:

| Length | GPIO_ID ₁ | ... | GPIO_ID _n |
|---------|----------------------|-----|----------------------|
| 1 Bytes | 1 Byte | | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

CMD_GPIO_LOCAL_READ_CNF block structure

Each **Block** has the following format:

| Length | GPIO_ID | Value |
|--------|---------|--------|
| 0x02 | 1 Byte | 1 Byte |

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 13.1

Value:

- if **Function** is output or input
0x00: The remote GPIO is LOW.
0x01: The remote GPIO is HIGH.
0xFF: Failed reading remote GPIO value.

9.6.8.1. Example: Read the values of local GPIOs

Read the value of the GPIOs with GPIO_ID **0x01** and **0x02** of the local device:

| Start signal | Command | Length | Block ₁ | CS |
|--------------|---------|-----------|-----------------------|------|
| 0x02 | 0x27 | 0x03 0x00 | 0x02 0x01 0x02 | 0x27 |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|--------|-----------------------|-----------------------|------|
| 0x02 | 0x67 | 0x07 0x00 | 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0x60 |

Successfully read the values of the local GPIOs with GPIO_ID **0x01** (GPIO is low) and **0x02** (GPIO is high).

9.6.9. CMD_GPIO_REMOTE_WRITECONFIG_IND

This command indicates that the remote device has written the free GPIOs of the radio module.



Please note that only the GPIOs are part of this message, that have been configured successfully. Failed attempts of GPIO configurations will not be indicated by this message.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0xA8 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

The **Block** structure is as defined in CMD_GPIO_REMOTE_WRITECONFIG_REQ block structure.

9.6.9.1. Example: Two GPIOs have been configured by the connected remote device to output high

| Start signal | Command | Length | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|----------------------------|----------------------------|----|
| 0x02 | 0xA8 | 0x08 0x00 | 0x03 0x01 0x02 0x01 | 0x03 0x02 0x02 0x01 | A1 |

The two GPIOs with ID **0x01** and **0x02** have been configured by the connected remote device to output high.

9.6.10. CMD_GPIO_REMOTE_WRITE_IND

This command indicates that the remote device has written the free GPIOs of the radio module.



Please note that only the GPIOs are part of this message, that have been updated successfully. Failed attempts of GPIO updates will not be indicated by this message.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0xA9 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

The **Block** structure is as defined in CMD_GPIO_LOCAL_READ_CNF block structure.

9.6.10.1. Example: GPIOs have been written via remote access

| Start signal | Command | Length | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|-----------------------|-----------------------|------|
| 0x02 | 0xA9 | 0x06 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0xAE |

The remote device has written the GPIOs with GPIO_ID **0x01** (GPIO is low) and **0x02** (GPIO is high).

9.6.11. CMD_GPIO_LOCAL_WRITE_IND

This command indicates that the GPIOs of the remote device have been written by its local host.



Please note that only the GPIOs are part of this message, that have been updated successfully. Failed attempts of GPIO updates will not be indicated by this message.

Format:

| Start signal | Command | Length | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|--------------------|-----|--------------------|--------|
| 0x02 | 0xA6 | 2 Bytes | x Bytes | | x Bytes | 1 Byte |

The **Block** is of structure as defined in CMD_GPIO_LOCAL_READ_CNF block structure .

9.6.11.1. Example: GPIOs of the remote device have been written by its local host

| Start signal | Command | Length | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|-----------------------|-----------------------|------|
| 0x02 | 0xA6 | 0x06 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0xA1 |

The GPIOs with GPIO_ID **0x01** (GPIO is low) and **0x02** (GPIO is high) of the radio module have been written by its local host.

9.7. Other messages

9.7.1. CMD_ERROR_IND

This indication is shown when the module entered an error state.

Format:

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|--------|
| 0x02 | 0xA2 | 0x01 0x00 | 1 Byte | 1 Byte |

Status:

0x01: UART_COMMUNICATION_ERROR The UART had a buffer overflow. Thus, UART TX and RX was aborted and UART has restarted. Please restart module if UART is still malfunctioning.

9.8. Message overview

| Start signal | CMD | Message name | Short description | Chapter |
|--------------|------|---------------------------------|--|---------|
| 0x02 | 0x00 | CMD_RESET_REQ | Reset the module | 9.4.2 |
| 0x02 | 0x01 | CMD_GETSTATE_REQ | Request the current module state | 9.4.1 |
| 0x02 | 0x02 | CMD_SLEEP_REQ | Go to sleep | 9.4.3 |
| 0x02 | 0x04 | CMD_DATA_REQ | Send data to the connected device | 9.2.1 |
| 0x02 | 0x07 | CMD_DISCONNECT_REQ | Close the connection | 9.1.4 |
| 0x02 | 0x0E | CMD_DELETEBONDS_REQ | Delete bonding information | 9.1.9 |
| 0x02 | 0x0F | CMD_GETBONDS_REQ | Read the MACs of bonded devices | 9.1.8 |
| 0x02 | 0x10 | CMD_GET_REQ | Read the module settings in flash | 9.3.2 |
| 0x02 | 0x11 | CMD_SET_REQ | Modify the module settings in flash | 9.3.1 |
| 0x02 | 0x1A | CMD_PHYUPDATE_REQ | Update the PHY | 9.1.6 |
| 0x02 | 0x1B | CMD_UARTDISABLE_REQ | Disable the UART | 9.4.6 |
| 0x02 | 0x1C | CMD_FACTORYRESET_REQ | Perform a factory reset | 9.4.5 |
| 0x02 | 0x1D | CMD_DTMSTART_REQ | Enable the direct test mode | 9.5.1 |
| 0x02 | 0x1E | CMD_DTM_REQ | Start/stop a test of the direct test mode | 9.5.2 |
| 0x02 | 0x25 | CMD_GPIO_LOCAL_WRITECONFIG_REQ | Configure the free GPIOs for remote control | 9.6.1 |
| 0x02 | 0x26 | CMD_GPIO_LOCAL_WRITE_REQ | Set the output value of a output GPIO of the current device | 9.6.7 |
| 0x02 | 0x27 | CMD_GPIO_LOCAL_READ_REQ | Read the value of a GPIO of the current device | 9.6.8 |
| 0x02 | 0x28 | CMD_GPIO_REMOTE_WRITECONFIG_REQ | Configure the free GPIOs of the remote device for remote control | 9.6.3 |
| 0x02 | 0x29 | CMD_GPIO_REMOTE_WRITE_REQ | Set the output value of a output GPIO of a remote device | 9.6.5 |
| 0x02 | 0x2A | CMD_GPIO_REMOTE_READ_REQ | Read the value of a GPIO of a remote device | 9.6.6 |
| 0x02 | 0x2B | CMD_GPIO_LOCAL_READCONFIG_REQ | Read the GPIO configuration | 9.6.2 |
| 0x02 | 0x2C | CMD_GPIO_REMOTE_READCONFIG_REQ | Read the GPIO configuration of the connected remote device | 9.6.4 |

| | | | | |
|------|------|----------------------------------|---|--------|
| 0x02 | 0x2D | CMD_ALLOWUNBONDEDCONNECTIONS_REQ | Temporarily allow the connection setup from unbonded peer devices | 9.1.10 |
|------|------|----------------------------------|---|--------|

Table 16: Message overview: Requests

| Start signal | CMD | Message name | Short description | Chapter |
|--------------|------|---------------------------------|---|---------|
| 0x02 | 0x40 | CMD_RESET_CNF | Reset request received | 9.4.2 |
| 0x02 | 0x41 | CMD_GETSTATE_CNF | Return the current module state | 9.4.1 |
| 0x02 | 0x42 | CMD_SLEEP_CNF | Sleep request received | 9.4.3 |
| 0x02 | 0x44 | CMD_DATA_CNF | Data transmission request received | 9.2.1 |
| 0x02 | 0x47 | CMD_DISCONNECT_CNF | Disconnection request received | 9.1.4 |
| 0x02 | 0x4E | CMD_DELETEBONDS_CNF | Deleted bonding information | 9.1.9 |
| 0x02 | 0x4F | CMD_GETBONDS_CNF | Return the MAC of all bonded devices | 9.1.8 |
| 0x02 | 0x50 | CMD_GET_CNF | Return the requested module flash settings | 9.3.2 |
| 0x02 | 0x51 | CMD_SET_CNF | Module flash settings have been modified | 9.3.1 |
| 0x02 | 0x5A | CMD_PHYUPDATE_CNF | Update Phy request received | 9.1.6 |
| 0x02 | 0x5B | CMD_UARTDISABLE_CNF | Disable UART request received | 9.4.6 |
| 0x02 | 0x5C | CMD_FACTORYRESET_CNF | Factory reset request received | 9.4.5 |
| 0x02 | 0x5D | CMD_DTMSTART_CNF | Enable the direct test mode now | 9.5.1 |
| 0x02 | 0x5E | CMD_DTM_CNF | Test of direct test mode started/stopped | 9.5.2 |
| 0x02 | 0x65 | CMD_GPIO_LOCAL_WRITECONFIG_CNF | Configuration of a local GPIO for remote control done | 9.6.1 |
| 0x02 | 0x66 | CMD_GPIO_LOCAL_WRITE_CNF | Output value of a local GPIO set | 9.6.7 |
| 0x02 | 0x67 | CMD_GPIO_LOCAL_READ_CNF | Value of a local GPIO read | 9.6.8 |
| 0x02 | 0x68 | CMD_GPIO_REMOTE_WRITECONFIG_CNF | Configuration of a remote GPIO for remote control done | 9.6.3 |
| 0x02 | 0x69 | CMD_GPIO_REMOTE_WRITE_CNF | Output value of a remote GPIO set | 9.6.5 |
| 0x02 | 0x6A | CMD_GPIO_REMOTE_READ_CNF | Value of a remote GPIO read | 9.6.6 |
| 0x02 | 0x6B | CMD_GPIO_LOCAL_READCONFIG_CNF | Returns the GPIO configuration | 9.6.2 |
| 0x02 | 0x6C | CMD_GPIO_REMOTE_READCONFIG_CNF | Returns the GPIO configuration of the connected remote device | 9.6.4 |

| | | | | |
|------|------|----------------------------------|---|--------|
| 0x02 | 0x6D | CMD_ALLOWUNBONDEDCONNECTIONS_CNF | Temporarily allowed the connection setup from unbonded peer devices | 9.1.10 |
|------|------|----------------------------------|---|--------|

Table 17: Message overview: Confirmations

| Start signal | CMD | Message name | Short description | Chapter |
|--------------|------|---------------------------------|---|---------|
| 0x02 | 0x82 | CMD_SLEEP_IND | State will be changed to ACTION_SLEEP | 9.4.4 |
| 0x02 | 0x84 | CMD_DATA_IND | Data has been received | 9.2.3 |
| 0x02 | 0x86 | CMD_CONNECT_IND | Connection established | 9.1.1 |
| 0x02 | 0x87 | CMD_DISCONNECT_IND | Disconnected | 9.1.5 |
| 0x02 | 0x88 | CMD_SECURITY_IND | Secured connection established | 9.1.2 |
| 0x02 | 0x9A | CMD_PHYUPDATE_IND | PHY has been updated | 9.1.7 |
| 0x02 | 0x9B | CMD_UARTENABLE_IND | UART was re-enabled | 9.4.7 |
| 0x02 | 0xA2 | CMD_ERROR_IND | Entered error state | 9.7.1 |
| 0x02 | 0xA6 | CMD_GPIO_LOCAL_WRITE_IND | Local host has written the GPIOs of the remote device | 9.6.11 |
| 0x02 | 0xA8 | CMD_GPIO_REMOTE_WRITECONFIG_IND | Remote device has configured the GPIOs of the module | 9.6.9 |
| 0x02 | 0xA9 | CMD_GPIO_REMOTE_WRITE_IND | Remote device has written the GPIOs of the module | 9.6.10 |
| 0x02 | 0xC4 | CMD_TXCOMPLETE_RSP | Data has been sent | 9.2.2 |
| 0x02 | 0xC6 | CMD_CHANNELOPEN_RSP | Channel open, data transmission possible | 9.1.3 |

Table 18: Message overview: Indications

10. User settings - Module configuration values

The settings described in this chapter are stored permanently in the module's flash memory. Depending on their corresponding permissions, their current values can be read out by the `CMD_GET_REQ` command or modified by the `CMD_SET_REQ` command. To do so the corresponding settings index is used, which can be found in the primary table of each setting description.



The validity of the specified parameters is not verified. Incorrect values can result in device malfunction.



After the modification of the non-volatile parameters, a reset will be necessary for the changes to be applied.

10.1. FS_DeviceInfo: Read the chip type and OS version

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------|--------------------|---------------|-------------|-----------------|
| 15 | FS_DeviceInfo | - | - | read | 12 |

This setting contains information about the chip type and the OS version. The value of `FS_DeviceInfo` is composed of the following 4 sub parameters (ordered by appearance in the response):

| OS version | Build code | Package variant | Chip ID |
|------------|------------|-----------------|---------|
| 2 Bytes | 4 Bytes | 2 Bytes | 4 Bytes |

OS version:

0x0126 : Softdevice S112 7.3.0.

Package variant:

0x2005: WLCSP

Chip ID:

0x00052805: nRF52805

| Packet variant | Flash size | RAM size |
|----------------|------------|----------|
| WLCSP | 196 kB | 24 kB |

Table 19: nRF52805 IC revision overview

10.1.1. Example 1

Request the device info of the module using CMD_GET_REQ with settings index 15

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x0F | 0x1C |

Response CMD_GET_CNF: Successfully read out the device info (with Byte order changed to MSB first):

OS version = 0x0126 (Softdevice S112 7.3.0)

Build code = 0x41414330 (AAC0)

Package variant = 0x2005 (WLCSP)

Chip ID = 0x00052805

Please note that LSB is transmitted first in case of parameters with more than 1 Byte length.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---|------|
| 0x02 | 0x50 | 0x0D 0x00 | 0x00 | 0x26 0x01 0x30 0x43 0x41 0x41 0x05 0x20 0x05 0x28 0x05 0x00 | 0x06 |

10.2. FS_FWVersion: Read the firmware version

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------|--------------------|---------------|-------------|-----------------|
| 1 | FS_FWVersion | - | - | read | 3 |

This setting contains the firmware version of the module.

10.2.1. Example 1

Request the firmware version of the module using CMD_GET_REQ with settings index 1

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x01 | 0x12 |

Response CMD_GET_CNF: Successfully read out the firmware version, for this example it is 0x000001 so "1.0.0" (with the parameter reverted to MSB first).

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------|------|
| 0x02 | 0x50 | 0x04 0x00 | 0x00 | 0x00 0x00 0x01 | 0x57 |

10.3. FS_MAC: Read the MAC address

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 3 | FS_MAC | - | - | read | 8 |

This setting contains the unique MAC address of the module.

10.3.1. Example 1

Request the MAC address of the module using CMD_GET_REQ with settings index 3

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x03 | 0x10 |

Response CMD_GET_CNF: Successfully read out the MAC address 0x55 0x93 0x19 0x6E 0x5B 0x87 0x01 0x38

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x50 | 0x09 0x00 | 0x00 | 0x55 0x93 0x19 0x6E 0x5B 0x87 0x01 0x38 | 0x0F |

10.4. FS_BTMAC: Modify the Bluetooth conform MAC address

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|-----------------|-------------|-----------------|
| 4 | FS_BTMAC | See description | See description | read/write | 6 |

This setting contains the Bluetooth® LE conform **public** MAC address of the module. It is used to identify the device on the radio interface.

By default, it is set to the MAC address bought by Würth Elektronik eiSos which consists of the Würth Elektronik eiSos MAC ID 0x0018DA followed by the FS_SerialNumber of the module.

Nevertheless, the FS_BTMAC can be set to any public MAC address, other than 0x0000000000 and 0xFFFFFFFFFFFF. For purchasing public MAC addresses, please refer to the IEEE standards association [10]. If set to 0xFFFFFFFFFFFF, the Proteus-e uses the default address mentioned above.

Please note that LSB is transmitted first in all commands.

10.4.1. Example 1

Set the Bluetooth®-conform MAC address of the module to 0x31 0x32 0x33 0x34 0x35 0x36 using CMD_SET_REQ with settings index 4

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|----------------------------------|------|
| 0x02 | 0x11 | 0x07 0x00 | 0x04 | 0x31 0x32 0x33 0x34 0x35 0x36 | 0x17 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.4.2. Example 2

Request the Bluetooth®-conform MAC address of the module using CMD_GET_REQ with settings index 4

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x04 | 0x17 |

Response CMD_GET_CNF: Successfully read out the Bluetooth® LE conform MAC address 0x11 0x00 0x00 0xDA 0x18 0x00.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------------------------|------|
| 0x02 | 0x50 | 0x07 0x00 | 0x00 | 0x11 0x00 0x00 0xDA 0x18 0x00 | 0x86 |

10.5. FS_SerialNumber: Read the serial number of the module

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------|--------------------|---------------|-------------|-----------------|
| 16 | FS_SerialNumber | - | - | read | 3 |

This setting contains the serial number of the module.

10.5.1. Example 1

Request the serial number of the module using CMD_GET_REQ with settings index 16

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x10 | 0x03 |

Response CMD_GET_CNF: Successfully read out the serial number, it is 0.0.11

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------|------|
| 0x02 | 0x50 | 0x04 0x00 | 0x00 | 0x11 0x00 0x00 | 0x57 |

10.6. RF_DeviceName: Modify the device name

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------|--------------------|---------------|-------------|-----------------|
| 2 | RF_DeviceName | See description | "Proteus-e" | read/write | 1-31 |



This parameter is using MSB first notation.

This parameter determines the name of the module, which is used in the advertising packets as well as in the Generic Access Profile (GAP). The permissible characters are in the range of 0x20 - 0x7E which are special characters (see ASCII table), alphabetic characters (a-z and A-Z), numbers (0-9) and whitespace (0x20).



The maximum size of the device name that fits into an advertising packet is 26 Bytes. Thus longer device names will be shortened to 26 Bytes and declared as "Shortened Local Name" in the advertising packet. The full device name is included in the GAP.



In case the device name is not changed from default value, the ASCII serial number of the module is appended.

10.6.1. Example 1

Set the device name of the module to 0x4D 0x4F 0x44 0x20 0x31 = "MOD 1" using CMD_SET_REQ with settings index 2.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|--------------------------|------|
| 0x02 | 0x11 | 0x06 0x00 | 0x02 | 0x4D 0x4F 0x44 0x20 0x31 | 0x40 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.6.2. Example 2

Request the device name of the module using CMD_GET_REQ with settings index 2:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x02 | 0x11 |

Response CMD_GET_CNF: Successfully read out the module name as 0x50 0x72 0x6F 0x74 0x65 0x75 0x73 0x2D 0x65 = "Proteus-e".

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x50 | 0x0A 0x00 | 0x00 | 0x50 0x72 0x6F 0x74 0x65 0x75 0x73 0x2D 0x65 | 0x4A |

10.7. RF_StaticPasskey: Modify the static passkey

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------|--------------------|---------------|-------------|-----------------|
| 18 | RF_StaticPasskey | See description | "123123" | read/write | 6 |

This setting determines the static pass key of the peripheral device used for authentication. If the static pass key security mode is enabled by the peripheral, this key must be entered in the central device.

The permissible characters are ranging from 0x30 to 0x39 which are ASCII numbers (0-9). This is due to the fact that mobile phones prefer numbers only for the passkey.

10.7.1. Example 1

Set the static pass key of the module to 0x31 0x32 0x33 0x34 0x35 0x36 = "123456" using CMD_SET_REQ with settings index 18

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|----------------------------------|------|
| 0x02 | 0x11 | 0x07 0x00 | 0x12 | 0x31 0x32 0x33 0x34 0x35 0x36 | 0x01 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.7.2. Example 2

Request the static pass key of the module using CMD_GET_REQ with settings index 18

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x12 | 0x01 |

Response CMD_GET_CNF: Successfully read out the key as 0x31 0x32 0x33 0x34 0x35 0x36 = "123456"

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-------------------------------|------|
| 0x02 | 0x50 | 0x07 0x00 | 0x00 | 0x31 0x32 0x33 0x34 0x35 0x36 | 0x52 |

10.8. RF_SecFlags: Modify the security settings

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 12 | RF_SecFlags | See description | 0 | read/write | 1 |

This 8-Bit field configures security settings of the module. Chapter 7.6 contains further information about secure connections.



When connecting from a foreign device to a Proteus-e, the peripheral (Proteus-e) determines the minimum security level needed for communication. So configure the RF_SecFlags of the peripheral to set the desired security level. When connecting from a Proteus-I,-II,-III to a Proteus-e, be sure that the same security mode is used.



When updating this user setting (like enabling bonding or changing the security mode) please remove all existing bonding data on both connection partners. On Proteus-e this can be achieved using the command `CMD_DELETEBONDS_REQ`.

| Bit no. | Description | | |
|---------|--|---------------------------|---|
| 2 : 0 | Security mode configuration. Depending on its value, different modes are chosen when setting up a secure connection. | | |
| | 0x0 | No security | Data is transmitted without authentication and encryption. |
| | 0x2 | Just works level 1.2 | Each time a connection is established, new random keys are exchanged in advance to use them for data encryption. This mode uses the "just works" method. |
| | 0x3 | Static pass key level 1.3 | For authentication, the RF_StaticPasskey is used. If the peripheral uses this method, the central device must enter the correct passkey to finalize the connection. |
| | others | | Reserved |
| 3 | SECFLAGS_BONDING_ENABLE: If this Bit is set, bonding is enabled when using one of the pairing methods. Bonding data of up to 12 devices will be stored in the flash. If bonding storage is full, the bonding information that has not been used for the longest period will be removed. | | |
| 4 | SECFLAGS_BONDEDCONNECTIONSONLY_ENABLE: If this Bit is set, only bonded peer devices are allowed to connect. All connection requests from any unbonded peer device are rejected. In case this restriction shall be disabled temporarily to setup a bonding to a new peer device, use the command CMD_ALLOWUNBONDEDCONNECTIONS_REQ, which temporarily disables this restriction. If this feature is enabled, the maximum number of bonded devices is reduced to 8. | | |
| 7 : 5 | Reserved | | |

Table 20: Security configuration flags

10.8.1. Example 1

Set the security flags to 0x0B, to use the static passkey pairing and with bonding enabled, using CMD_SET_REQ with settings index 12

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x0C | 0x0B | 0x16 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.8.2. Example 2

Request the security flags of the module using CMD_GET_REQ with settings index 12

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x0C | 0x1F |

Response CMD_GET_CNF: Successfully read out the value 2, which means that the just works pairing mode is enabled.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x02 | 0x52 |

10.9. RF_ScanResponseData: Modify the content of the scan response packet

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------|--------------------|---------------|-------------|-----------------|
| 14 | RF_ScanResponseData | See description | None | read/write | 0-31 |

The standard content of the scan response packet of the Proteus-e is automatically defined. This setting allows to put user content in the scan response packet. The value of this user setting is the raw data which is placed without modification in the scan response packet, after the standard content has been removed before.



Please check chapter Advertising of custom data for details and restrictions before modifying this user setting.



Please ensure that the raw data is compliant to the Bluetooth® specification [1] chapter "ADVERTISING AND SCAN RESPONSE DATA FORMAT". Otherwise it can result in malfunctioning.



To use the standard content of the scan response packet again, please set this user setting RF_ScanResponseData to a value with zero length.

10.9.1. Example 1

Set the custom data of the scan response packet to:

- TX power is 4 (0x02 **0x0A** 0x04)

using CMD_SET_REQ with settings index 14

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------------------|------|
| 0x02 | 0x11 | 0x04 0x00 | 0x0E | 0x02 0x0A 0x04 | 0x15 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.9.2. Example 2

Request the custom scan response data of the module using CMD_GET_REQ with settings index 14

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x0E | 0x1D |

Response CMD_GET_CNF:Successfully read out the content as:

- TX power is 4 (0x02 0x0A 0x04)

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------|------|
| 0x02 | 0x50 | 0x04 0x00 | 0x00 | 0x02 0x0A 0x04 | 0x5A |

10.10. RF_AdvertisingData: Modify the content of the advertising packet

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------------|--------------------|---------------|-------------|-----------------|
| 13 | RF_AdvertisingData | See description | None | read/write | 0-31 |

The standard content of the advertising packet of the Proteus-e is automatically defined. This setting allows to put user content in the advertising packet. The value of this user setting is the raw data which is placed without modification in the advertising packet, after the standard content has been removed before.



Please check chapter Advertising of custom data for details and restrictions before modifying this user setting.



Please ensure that the raw data is compliant to the Bluetooth® specification [1] chapter "ADVERTISING AND SCAN RESPONSE DATA FORMAT". Otherwise it can result in malfunctioning.



To use the standard content of the advertising packet again, please set this user setting RF_AdvertisingData to a value with zero length.

10.10.1. Example 1

Set the custom data of the advertising packet to:

- full device name is "Hello" (0x06 **0x09** 0x48 0x65 0x6C 0x6C 0x6F)
- TX power is 4 (0x02 **0x0A** 0x04)

using CMD_SET_REQ with settings index 13

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|---|------|
| 0x02 | 0x11 | 0x0B 0x00 | 0x0D | 0x06 0x09 0x48 0x65 0x6C 0x6C 0x6F 0x02 0x0A 0x04 | 0x54 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.10.2. Example 2

Request the custom advertising data of the module using CMD_GET_REQ with settings index 13

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x0D | 0x1E |

Response CMD_GET_CNF:Successfully read out the content as:

- full device name is "Hello" (0x06 0x09 0x48 0x65 0x6C 0x6C 0x6F)
- TX power is 4 (0x02 0x0A 0x04)

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x50 | 0x0B 0x00 | 0x00 | 0x06 0x09 0x48 0x65 0x6C 0x6C 0x6F 0x02 0x0A 0x04 | 0x18 |

10.11. RF_AdvertisingTimeout: Modify the advertising timeout

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------------|--------------------------|---------------|-------------|-----------------|
| 7 | RF_AdvertisingTimeout | 0 (infinite), 1 - 650 | 0 | read/write | 2 |

This parameter defines the time in seconds after which the advertising of the module stops. If no peer connects before this timeout, advertising stops and the module goes to sleep mode. If the RF_AdvertisingTimeout is set to 0, the module advertises infinitely.



To ensure that the module sends a sufficient amount of advertising packets per RF_AdvertisingTimeout, please also check the RF_AdvertisingInterval parameter, which defines the frequency of advertising packets.

10.11.1. Example 1

Set the advertising timeout parameter to 0x00 0xB4 (180s) using CMD_SET_REQ with settings index 7.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x07 | 0xB4 0x00 | 0xA3 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.11.2. Example 2

Request the advertising timeout of the module using CMD_GET_REQ with settings index 7

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x07 | 0x14 |

Response CMD_GET_CNF: Successfully read out the value 0x00 0x00 = 0s, which indicates indefinite advertising.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x51 |

10.12. RF_AdvertisingInterval: Modify the advertising interval

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------------|--------------------|---------------|-------------|-----------------|
| 9 | RF_AdvertisingInterval | 20 - 10240 | 40 | read/write | 2 |

The RF_AdvertisingInterval defines how often advertising packets are transmitted. The value is the interval in milliseconds.

The choice of the RF_AdvertisingInterval primarily affects the latency of device detection on air as well as the current consumption. A lower value of the RF_AdvertisingInterval results in a shorter pause between the advertising packets. Thus the radio module can be detected earlier, but also needs more power.

10.12.1. Example 1

Set the advertising interval to 100 ms using CMD_SET_REQ with settings index 9.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x09 | 0x64 0x00 | 0x7D |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.12.2. Example 2

Request the advertising interval of the module using CMD_GET_REQ with settings index 9

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x09 | 0x1A |

Response CMD_GET_CNF: Successfully read out the value 0x0028 (40 ms).

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x28 0x00 | 0x79 |

10.13. RF_ConnectionInterval: Modify the connection interval

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------------|--------------------|---------------|-------------|-----------------|
| 8 | RF_ConnectionInterval | See description | 15,50 | read/write | 4 |

The user setting `RF_ConnectionInterval` defines the minimum and maximum connection interval, which is used to negotiate the connection interval during connection setup.

The 4 byte value of the user setting `RF_ConnectionInterval` consists of the 2 byte value of the minimum connection interval, followed by the 2 byte value of the maximum connection interval (see the examples below).

The value for the minimum connection interval, and the value for the maximum connection interval must be within 8 ms and 4000 ms. Only settings are accepted where the minimum connection interval is lower or equal to the maximum connection interval.

Further information:

- The minimum and maximum connection interval parameters specify the borders of the connection interval as determined in the negotiation procedure between the central and the peripheral during connection setup. The connection interval defines the frequency of communication during connection setup and data transmission. The lower the connection interval is, the more frequently the connected devices communicate with each other and thus the more power is consumed.

If a Bluetooth® LE device (e.g. a smart phone) connects as central to a Proteus-e module (peripheral) and the connection interval settings do not coincide, the Proteus-e requests the smart phone to accept its settings after 5 s. If the cell phone does not accept the settings, it will be requested a further 3 times with a delay of 10 s. If the peripheral's settings requests have been rejected in all cases the connection will be shut down. If the smart phone itself requests to update the connection interval of the Proteus-e, the module accepts the request.



Please ensure that all members (Proteus-e, cell phones and other Bluetooth® LE devices) of a network use the same connection timing parameters to avoid connection problems and changes of the connection interval during an opened connection.



Please check the minimum connection interval that is supported by iOS is 15 ms. Furthermore the minimum connection interval for Apple devices shall be a multiple of 15 ms! Please refer to Apple's Bluetooth® LE developer guide for best practices.

10.13.1. Example 1

Set the RF_ConnectionInterval to 16-40 ms (0x0010-0x0028) using CMD_SET_REQ with settings index 8.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|---------------------|------|
| 0x02 | 0x11 | 0x05 0x00 | 0x08 | 0x10 0x00 0x28 0x00 | 0x26 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.13.2. Example 2

Request the minimum and maximum connection interval of the module using CMD_GET_REQ with settings index 8

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x08 | 0x1B |

Response CMD_GET_CNF: Successfully read out the value 20-75 ms (0x0014-0x004B).

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---------------------|------|
| 0x02 | 0x50 | 0x05 0x00 | 0x00 | 0x14 0x00 0x4B 0x00 | 0x08 |

10.14. RF_TXPower: Modify the output power

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 17 | RF_TXPower | See description | 4 | read/write | 1 |

This setting determines the output power in dBm of the module. The value has to be entered in hexadecimal and as two's complement. The permissible values are listed in the following table.

| Permissible values | | | | | |
|-------------------------------|------|------|------|------|------|
| Decimal [dBm] | -40 | -20 | -16 | -12 | -8 |
| Two's complement, hexadecimal | 0xD8 | 0xEC | 0xF0 | 0xF4 | 0xF8 |

| Decimal [dBm] | -4 | 0 | 3 | 4 | |
|-------------------------------|------|------|------|------|--|
| Two's complement, hexadecimal | 0xFC | 0x00 | 0x03 | 0x04 | |



Please note that this setting defines the TX power value the radio chip provides. The radio signal will be damped on its way along the RF path and the antenna. For maximum output signal at the *RF* pin or integrated antenna, please refer to the chapter *Radio characteristics*.

10.14.1. Example 1

Set the output power of the module to -8 dBm, which is 0xF8 in two's complement notation, using CMD_SET_REQ with settings index 17

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x11 | 0xF8 | 0xF8 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.14.2. Example 2

Request the output power of the module using CMD_GET_REQ with settings index 17

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x11 | 0x02 |

Response CMD_GET_CNF: Successfully read out the value 0x04 = 4dBm

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x04 | 0x54 |

10.15. RF_SPPBaseUUID: Configure the SPP base UUID

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|----------------|--------------------|--|-------------|-----------------|
| 26 | RF_SPPBaseUUID | See description | 0x6E400000C352 11E5953D0002 A5D5C51B | read/write | 16 |

Set the base UUID of the SPP-like profile. For more information about the UUID definition, please refer to chapter 14.1.



Please note that the UUID 0xFB349B5F8000008000100000xxxx0000 (0000xxxx-0000-1000-8000-00805f9b34fb) is reserved for 16 Bit UUIDs and must not be used here.

10.15.1. Example 1

Set the base UUID to 0xEFEEEEDEC-EBEA-E9E8-E7E6-E5E4E3E2E1E0 using CMD_SET_REQ with settings index 26

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|--|------|
| 0x02 | 0x11 | 0x11 0x00 | 0x1A | 0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9 0xEA 0xEB 0xEC 0xED 0xEE 0xEF | 0x18 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.15.2. Example 2

Request the base UUID of the module using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x1A | 0x09 |

Response CMD_GET_CNF: Successfully read out the value 0x6E400000-C352-11E5-953D-0002A5D5C51B.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---|------|
| 0x02 | 0x50 | 0x11 0x00 | 0x00 | 0x1B 0xC5 0xD5 0xA5 0x02 0x00 0x3D 0x95 0xE5 0x11 0x52 0xC3 0x00 0x00 0x40 0x6E | 0x0C |

10.16. RF_SPPServiceUUID: Configure the SPP service UUID

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------------|--------------------|---------------|-------------|-----------------|
| 32 | RF_SPPServiceUUID | See description | 0x0001 | read/write | 2 |

Set the service UUID of the SPP-like profile. For more information about the UUID definition, please refer to chapter 14.1.

The service UUID can be any value, but must be different from RF_SPPTXUUID and RF_SPPRXUUID.

10.16.1. Example 1

Set the service UUID to 0x1122 using CMD_SET_REQ with settings index 32

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x20 | 0x22 0x11 | 0x03 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.16.2. Example 2

Request the service UUID of the module using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x20 | 0x33 |

Response CMD_GET_CNF: Successfully read out the value 0x1234.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x34 0x12 | 0x77 |

10.17. RF_SPPRXUUID: Configure the SPP RX UUID

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------|--------------------|---------------|-------------|-----------------|
| 33 | RF_SPPRXUUID | See description | 0x0002 | read/write | 2 |

Set the RX UUID of the SPP-like profile. This characteristic has the function to transmit data from the connected remote peer to the radio module via write command. For more information about the UUID definition, please refer to chapter 14.1.



The RX UUID can be any value, but must be different from RF_SPPServiceUUID and RF_SPPTXUUID.

10.17.1. Example 1

Set the RX UUID to 0x1122 using CMD_SET_REQ with settings index 33

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x21 | 0x22 0x11 | 0x02 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.17.2. Example 2

Request the service UUID of the module using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x21 | 0x32 |

Response CMD_GET_CNF: Successfully read out the value 0x1234.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x34 0x12 | 0x77 |

10.18. RF_SPPTXUUID: Configure the SPP TX UUID

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------|--------------------|---------------|-------------|-----------------|
| 34 | RF_SPPTXUUID | See description | 0x0003 | read/write | 2 |

Set the TX UUID of the SPP-like profile. This characteristic has the function to transmit data from the radio module to the connected remote peer via notification. For more information about the UUID definition, please refer to chapter 14.1.



The TX UUID can be any value, but must be different from RF_SPPServiceUUID and RF_SPPRXUUID.

10.18.1. Example 1

Set the TX UUID to 0x1122 using CMD_SET_REQ with settings index 34

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x22 | 0x22 0x11 | 0x01 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.18.2. Example 2

Request the service UUID of the module using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x22 | 0x31 |

Response CMD_GET_CNF: Successfully read out the value 0x1234.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x34 0x12 | 0x77 |

10.19. RF_Appearance: Configure the appearance of the device

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------|--------------------|---------------|-------------|-----------------|
| 25 | RF_Appearance | 0-65535 | 0 | read/write | 2 |

The user setting RF_Appearance specifies the appearance of the Bluetooth® devices. It's a 2 Bytes field defined by the Bluetooth® SIG. Please check the Bluetooth® Core Specification:Core Specification Supplement, Part A, section 1.12 [1] for permissible values.

10.19.1. Example 1

Set the appearance to "Generic computer" (0x0080) using CMD_SET_REQ with settings index 25

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x19 | 0x80 0x00 | 0x89 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.19.2. Example 2

Request the RF_Appearance using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x19 | 0x0A |

Response CMD_GET_CNF: Successfully read out the value 0x0000, meaning that the appearance is unknown.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x51 |

10.20. UART_ConfigIndex: Modify the UART speed

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------|--------------------|---------------|-------------|-----------------|
| 11 | UART_ConfigIndex | See description | 22 | read/write | 1 |

This parameter defines the baud rate used by the module's UART. The permissible values are listed in the following table. If flow control is enabled the pins */RTS* and */CTS* are used.

| UART_ConfigIndex | Rate [Baud] | Real rate [Baud] | Flow control | Parity |
|------------------|-------------|------------------|--------------|--------|
| 0 | 1200 | 1205 | no | none |
| 1 | 1200 | 1205 | yes | none |
| 2 | 2400 | 2396 | no | none |
| 3 | 2400 | 2396 | yes | none |
| 4 | 4800 | 4808 | no | none |
| 5 | 4800 | 4808 | yes | none |
| 6 | 9600 | 9598 | no | none |
| 7 | 9600 | 9598 | yes | none |
| 8 | 14400 | 14414 | no | none |
| 9 | 14400 | 14414 | yes | none |
| 10 | 19200 | 19208 | no | none |
| 11 | 19200 | 19208 | yes | none |
| 12 | 28800 | 28829 | no | none |
| 13 | 28800 | 28829 | yes | none |
| 14 | 38400 | 38462 | no | none |
| 15 | 38400 | 38462 | yes | none |
| 16 | 56000 | 55944 | no | none |
| 17 | 56000 | 55944 | yes | none |
| 18 | 57600 | 57762 | no | none |
| 19 | 57600 | 57762 | yes | none |
| 20 | 76800 | 76923 | no | none |
| 21 | 76800 | 76923 | yes | none |
| 22 | 115200 | 115942 | no | none |
| 23 | 115200 | 115942 | yes | none |
| 25 | 230400 | 231884 | yes | none |
| 27 | 250000 | 250000 | yes | none |
| 29 | 460800 | 470588 | yes | none |
| 31 | 921600 | 941176 | yes | none |

| | | | | |
|--------|----------|---------|-----|------|
| 33 | 1000000 | 1000000 | yes | none |
| 64 | 1200 | 1205 | no | even |
| 65 | 1200 | 1205 | yes | even |
| 66 | 2400 | 2396 | no | even |
| 67 | 2400 | 2396 | yes | even |
| 68 | 4800 | 4808 | no | even |
| 69 | 4800 | 4808 | yes | even |
| 70 | 9600 | 9598 | no | even |
| 71 | 9600 | 9598 | yes | even |
| 72 | 14400 | 14414 | no | even |
| 73 | 14400 | 14414 | yes | even |
| 74 | 19200 | 19208 | no | even |
| 75 | 19200 | 19208 | yes | even |
| 76 | 28800 | 28829 | no | even |
| 77 | 28800 | 28829 | yes | even |
| 78 | 38400 | 38462 | no | even |
| 79 | 38400 | 38462 | yes | even |
| 80 | 56000 | 55944 | no | even |
| 81 | 56000 | 55944 | yes | even |
| 82 | 57600 | 57762 | no | even |
| 83 | 57600 | 57762 | yes | even |
| 84 | 76800 | 76923 | no | even |
| 85 | 76800 | 76923 | yes | even |
| 86 | 115200 | 115942 | no | even |
| 87 | 115200 | 115942 | yes | even |
| 89 | 230400 | 231884 | yes | even |
| 91 | 250000 | 250000 | yes | even |
| 93 | 460800 | 470588 | yes | even |
| 95 | 921600 | 941176 | yes | even |
| 97 | 1000000 | 1000000 | yes | even |
| Others | Reserved | | | |



After changing the baud rate using the CMD_SET_REQ the module restarts using the new baud rate. Therefore don't forget to update the baud rate of the connected host to be able to further use the module's UART.



Please note that due to the radio activity of the chip, single Bytes on the UART can get lost, when using a very fast UART data rate. To avoid losing single bytes, please enable the UART flow control and implement it on the host accordingly.

10.20.1. Example 1

Set the baud rate index to 0x1F (921600 Baud with flow control and parity none) using CMD_SET_REQ with settings index 11

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x0B | 0x1F | 0x05 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.20.2. Example 2

Request the baud rate index of the module using CMD_GET_REQ with settings index 11

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x0B | 0x18 |

Response CMD_GET_CNF: Successfully read out the value 0x16, which equals 115200 Baud without flow control and parity none.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x16 | 0x46 |

10.21. UART_TransparentTimeout: Modify the transparent mode timeout

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------------------|--------------------|---------------|-------------|-----------------|
| 36 | UART_TransparentTimeout | 2 - 255 | 10 | read/write | 1 |

This parameter defines the time in milliseconds after which the module starts transmitting the received UART data via radio, if no new data has been received for that time.

10.21.1. Example 1

Set the timeout parameter to 0x64 (100 ms) using CMD_SET_REQ with settings index 36.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x24 | 0x64 | 0x51 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.21.2. Example 2

Request the timeout of the module using CMD_GET_REQ with settings index 36

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x24 | 0x37 |

Response CMD_GET_CNF: Successfully read out the value 0x0A = 10 ms.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x0A | 0x5A |

10.22. UART_TransparentMaxPayload: Modify the transparent mode max payload

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|----------------------------|--------------------|---------------|-------------|-----------------|
| 37 | UART_TransparentMaxPayload | 1 - 247 | 243 | read/write | 2 |

This parameter defines the number of bytes after which the module starts the radio transmission.

10.22.1. Example 1

Set the maximum payload parameter to 0x64 (100) using CMD_SET_REQ with settings index 37.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x25 | 0x00 0x64 | 0x51 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.22.2. Example 2

Request the maximum payload of the module using CMD_GET_REQ with settings index 37.

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x25 | 0x36 |

Response CMD_GET_CNF: Successfully read out the value 0x0A = 10.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x0A 0x00 | 0x5B |

10.23. UART_TransparentETX: Modify the transparent ETX characters

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------|--------------------|---------------|-------------|-----------------|
| 5 | UART_TransparentETX | - | 0x0D0A | read/write | 2 |

Depending on the parameter UART_TransparentETXConfig, this parameter defines the ETX characters used to trigger radio transmission, when the specified ETX characters have been received and detected on the UART.

10.23.1. Example 1

Set the ETX to 0x1122 using CMD_SET_REQ with settings index 5.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x05 | 0x11 0x22 | 0x26 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.23.2. Example 2

Request the ETX characters of the module using CMD_GET_REQ with settings index 5.

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x05 | 0x16 |

Response CMD_GET_CNF: Successfully read out the value 0x0D 0x0A.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x0D 0x0A | 0x56 |

10.24. UART_TransparentETXConfig: Configure the usage of the ETX in transparent mode

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------------|--------------------|---------------|-------------|-----------------|
| 6 | UART_TransparentETXConfig | See description | 0 | read/write | 1 |

The user setting UART_TransparentETXConfig specifies which ETX triggers are used in transparent mode.

| Bit no. | Description |
|---------|--|
| 1 : 0 | Configuration mask |
| | 0x0 No ETX ETX trigger is disabled. |
| | 0x1 1 ETX One ETX character will be used. |
| | 0x2 2 ETX Two ETX characters will be used. |
| 2 | TRANSP_REMOVE_ETX: If this Bit is set, the ETX characters will be removed from the data stream before transmitting it via radio. |
| 3 | Reserved |

10.24.1. Example 1

Enable the 2 character ETX using CMD_SET_REQ with settings index 6

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x06 | 0x02 | 0x15 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command | Length | Status | CS |
|--------------|---------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.24.2. Example 2

Request the UART_TransparentETXConfig using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x06 | 0x15 |

Response CMD_GET_CNF: Successfully read out the value 0x06, meaning that a 2 character ETX trigger is enabled, and the ETX characters are removed from byte stream.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x06 | 0x56 |

10.25. CFG_Flags: Configure the module

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 28 | CFG_Flags | See description | 8 | read/write | 2 |

The user setting CFG_Flags specifies various module features.

| Bit no. | Name | Description |
|---------|--------------------|---|
| 0-1 | Reserved | Reserved. |
| 2 | GPIO remote config | Set this Bit to 1 to block the GPIO configuration via remote device. |
| 3 | DCDC enable | Set this Bit to 1 to use the internal DCDC instead of LDO. |
| 4 | Disconnect enable | Set this Bit to disconnect in case the central is forcing the use of invalid connection parameters. |
| 5-15 | Reserved | Reserved. |

10.25.1. Example 1

Block the GPIO configuration via remote and enable the DCDC access using CMD_SET_REQ with settings index 28

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x1C | 0x0C 0x00 | 0x00 |

Response CMD_SET_CNF: Successfully modified the setting.

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|------|
| 0x02 | 0x51 | 0x01 0x00 | 0x00 | 0x52 |

10.25.2. Example 2

Request the CFG_Flags using CMD_GET_REQ:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x1C | 0x0F |

Response CMD_GET_CNF: Successfully read out the value 0x00, meaning that all of the specified features are disabled.

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x00 0x00 | 0x51 |

| Settings index | Designation | Summary | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------------|---|-------------------------|-----------------|-------------|-----------------|
| 1 | FS_FWVersion | Version of the firmware | - | - | read | 3 |
| 2 | RF_DeviceName | Name of the module | See description | "Proteus-e" | read/write | 1-31 |
| 3 | FS_MAC | MAC address of the module | - | - | read | 6 |
| 4 | FS_BTMAC | Bluetooth® LE conform MAC address of the module | See description | See description | read/write | 6 |
| 5 | UART_TransparentETX | ETX characters | See description | 0x0D0A | read/write | 2 |
| 6 | UART_TransparentETXConfig | ETX configuration | See description | 0 | read/write | 1 |
| 7 | RF_AdvertisingTimeout | Time [s] after advertising stops. LSB first | 0 (infinite), 1 - 65535 | 0 | read/write | 2 |
| 8 | RF_ConnectionInterval | Connection interval | See description | 15,50 | read/write | 4 |
| 9 | RF_AdvertisingInterval | Advertising interval | 20-10240 | 40 | read/write | 2 |
| 11 | UART_ConfigIndex | Baud rate of the UART | See description | 22 | read/write | 1 |
| 12 | RF_SecFlags | Security settings of the module | See description | 0 | read/write | 1 |
| 13 | RF_AdvertisingData | Custom advertising packet data | See description | - | read/write | 0-31 |
| 14 | RF_ScanResponseData | Custom scan response packet data | See description | - | read/write | 0-31 |
| 15 | FS_DeviceInfo | Information about the chip | - | - | read | 12 |
| 16 | FS_SerialNumber | Serial number of the module | - | - | read | 3 |
| 17 | RF_TXPower | Output power [dBm] Two's complement | See description | 4 | read/write | 1 |
| 18 | RF_StaticPasskey | 6 digit pass key | See description | "123123" | read/write | 6 |
| 25 | RF_Appearance | Appearance | 0-65535 | 0 | read/write | 2 |
| 26 | RF_SPPBaseUUID | Base UUID of the SPP-like profile | See description | See description | read/write | 16 |
| 28 | CFG_Flags | CFG Flags | See description | 8 | read/write | 2 |

| | | | | | | |
|----|----------------------------|--------------------------------------|-----------------|--------|------------|---|
| 32 | RF_SPPServiceUUID | Service UUID of the SPP-like profile | See description | 0x0001 | read/write | 2 |
| 33 | RF_SPPRXUUID | RX UUID of the SPP-like profile | See description | 0x0002 | read/write | 2 |
| 34 | RF_SPPTXUUID | TX UUID of the SPP-like profile | See description | 0x0003 | read/write | 2 |
| 36 | UART_TransparentTimeout | Transparent timeout | 2 - 255 | 10 | read/write | 1 |
| 37 | UART_TransparentMaxPayload | Transparent payload | 1 - 247 | 243 | read/write | 2 |

Table 22: Table of user settings

| Settings index | Designation | Summary | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------|----------------------------------|--------------------|---------------|-------------|-----------------|
| 13 | RF_AdvertisingData | Custom advertising packet data | See description | - | read/write | 0-31 |
| 14 | RF_ScanResponseData | Custom scan response packet data | See description | - | read/write | 0-31 |

Table 23: Table of runtime settings

11. Timing parameters

11.1. Reset and sleep

After power-up, resetting the module or waking the module from sleep a `CMD_GETSTATE_CNF` is sent to the serial interface as soon as the module is ready for operation.

| Description | Typ. | Unit |
|-------------------------|------|------|
| Ready after reset/sleep | 12 | ms |

11.2. Bluetooth LE timing parameters

The timing parameters for sending advertising packets are determined by the user settings `RF_AdvertisingInterval` and `RF_AdvertisingTimeout`. Furthermore, the user setting `RF_ConnectionInterval` allows to configure the timing parameters used during connection setup and data transmission.

11.3. Connection establishment

The time needed to establish a connection sums up as the time needed to detect the selected peripheral on air and the time needed for connection parameter negotiation and service discovery.

Peripheral detection To establish a connection, the initiating device (central, e.g. smart phone) waits for an advertising packet, which was sent by the peripheral (Proteus-e) to which it wants to connect to. As soon as such an advertising packet has been received, the central sends a connection request to the chosen peripheral. The time needed to receive this advertising packet strongly depends on the advertising interval (see `RF_AdvertisingInterval`) of the peripheral as well as on the scan interval and scan window of the central.

Connection parameter negotiation After the connection request has been sent the central and peripheral negotiate the timing and security parameters of the connection. To finish this procedure and discover the services of the peripheral, several messages have to be sent, whereby only one is sent per connection interval (see `RF_ConnectionInterval`).

| Connection type | Estimated number of exchanged messages | Negotiation time for a connection interval of 50 ms |
|---|--|---|
| Unsecured connection | 12-14 | 600-700 ms |
| Secured connection using the pairing method | 22-24 | 1100-1200 ms |
| Secured connection to already bonded device | 19-20 | 950-1000 ms |

Knowing the connection interval and the number of messages that will be sent, the time necessary to setup a connection can be estimated by multiplying the number of messages with the connection interval.

11.4. Connection based data transmission

After connection has been setup, data can be transmitted using the `CMD_DATA_REQ`. It buffers the data in the module and sends it with the next connection interval event. As soon as the data has been transmitted successfully, a `CMD_TXCOMPLETE_RSP` is returned to the host. The time needed for this coincides with the connection interval that was negotiated during connection setup. The `RF_ConnectionInterval` parameter defines the minimum and maximum connection interval, which is supported by the module.

The following image shows the command sequence when sending data:

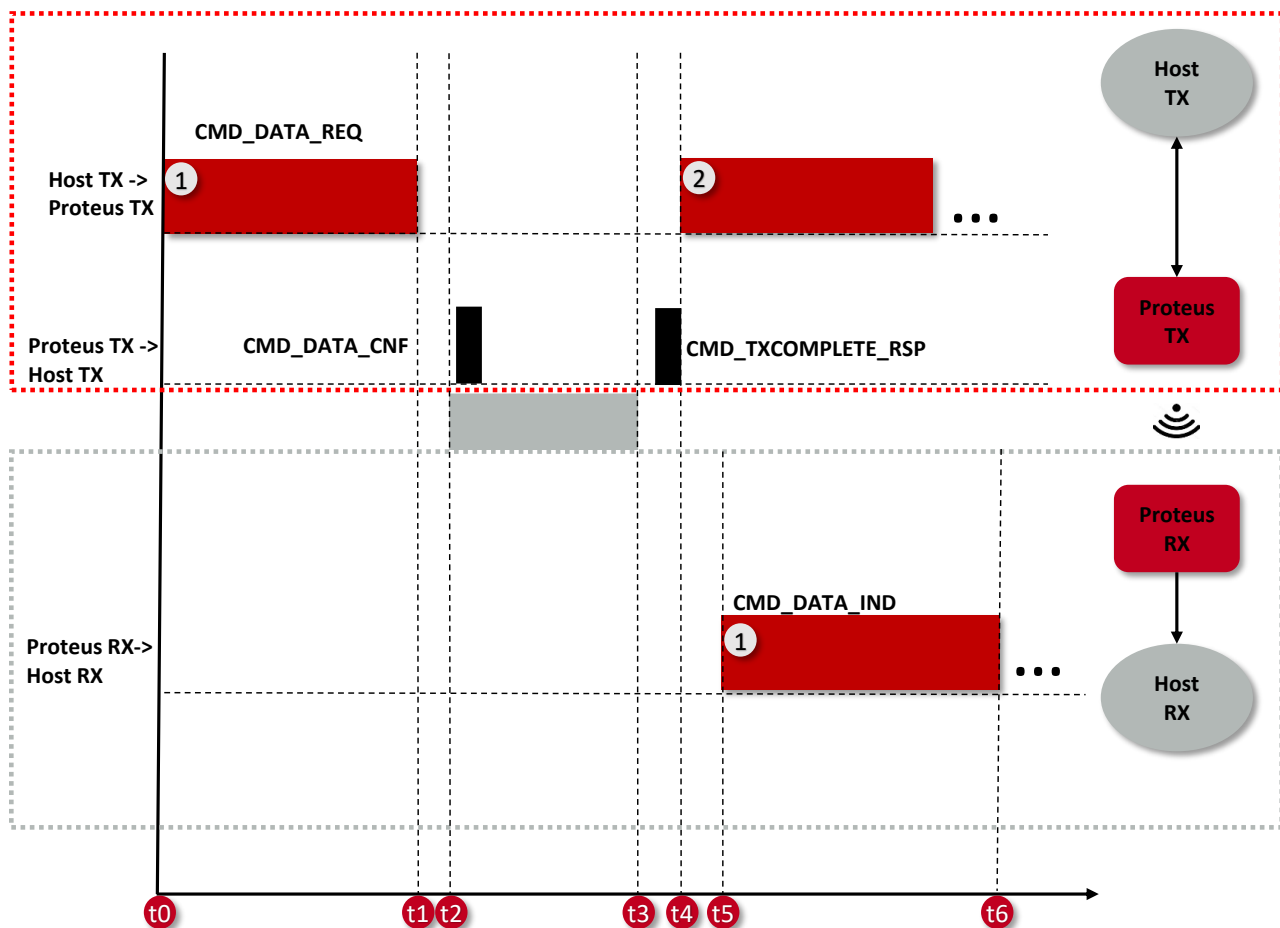


Figure 14: Command sequence when transmitting data

11.4.1. Maximum data throughput

The following table contains the measured maximum throughput values for user payload. The test setup is as follows:

- A Proteus-e radio module and a Proteus-III radio module connected to a fast micro controller (STM32 on NUCLEO-L476RG)
- Radio mode as specified in the table below (1 Mbit/s or 2 Mbit/s)
- Fastest connection interval of 8 ms (`RF_ConnectionInterval` equals 8-8ms)
- Fastest UART baud rate of 1000000 Baud (`UART_ConfigIndex` equals 33) with HW flow control
- 243 bytes per `CMD_DATA_REQ`

| Radio mode | t1-t0 [ms] (UART RX) | t4-t0 [ms] (Host TX Period) | t6-t5 [ms] (UART TX) | t6-t0 [ms] (End-to-end delay) | 243/(t4-t0) [kByte/s] (Throughput) |
|------------|-------------------------|--------------------------------|-------------------------|----------------------------------|--|
| 1 Mb/s | 3.85 | 22.5 | - | - | 10.8 |
| 2 Mb/s | 3.85 | 15 | - | - | 16.2 |

Table 24: Maximum throughput timings, packet error rate = 0%



Please note that data transmission to/from smart devices typically do not achieve this speed due to latency caused by the smart device and its software and apps or even missing hardware features such as Bluetooth® LE 5.0 full feature support.

12. Transparent mode

The Proteus-e implements a new feature that allows the easy integration of the Proteus-e Bluetooth® LE module to an already existing host. The transparent mode offers a plug and play installation without previous configuration of the Proteus-e. It is tailored for easy communication with mobile Bluetooth® LE devices like smart phones.

The transparent mode is a special operation mode, that uses the configuration of the command mode described in the previous chapters. It has to be enabled during the module start-up and contains the following key features:

- Automatic UART enabling: During advertising the UART is disabled. As soon as another Bluetooth® LE enabled device connects to it, the UART of the Proteus-e is enabled, the *LED_1* pin shows that the channel is open and bidirectional data transmission can start. As soon as the connection is closed, the UART is disabled.
- Transparent UART interface: The serial interface of the Proteus-e is no longer driven by commands. This means, when the UART of the module is enabled, all data sent to the UART is transmitted by the Proteus-e to the connected Bluetooth® LE enabled device. On the other hand, all data received by radio is sent from the Proteus-e to the connected host without additional protocol data. Since the commands of the command interface are no longer valid, a Proteus-e cannot be configured when running in transparent mode.

12.1. Reasons to use the transparent mode

The Proteus-e transparent mode equips custom applications with a Bluetooth® LE interface (to be accessible by other Bluetooth® LE devices) without installation effort. Due to the transparent UART interface, data can be exchanged without additional protocol. In addition to that, the transparent mode allows an energy efficient operation of the Bluetooth® LE interface, since the UART is only enabled when it is really used.

12.2. How to use the transparent mode

The transparent mode is enabled, when a high signal is present on the *MODE_1* pin during device start-up.

No configuration of the module is needed for this operating mode. The module shall be set to factory settings if reconfigured before so it uses the default user settings.

If changing the module's configuration is required (e.g. when another UART baud rate needs to be chosen), the module has to be started in command mode and the *CMD_SET_REQ* may be used to update the user settings.

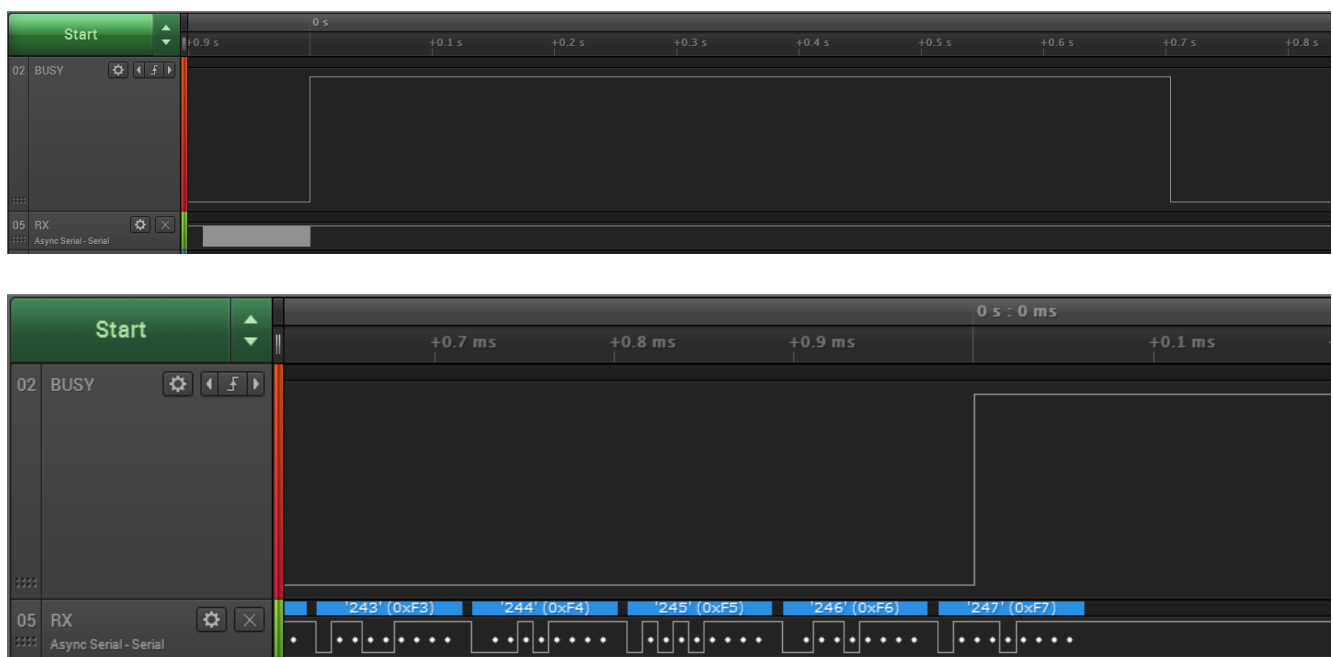
Any user setting of the command mode will also apply to the transparent mode.

12.3. More information

12.3.1. UART

- The data sent to the UART is buffered in the Proteus-e. There are different triggers, when the received UART data is transmitted via radio interface:

- If the number of bytes exceeds the `UART_TransparentMaxPayload`.
 - If no data has been received on the UART for a time between one and two times `UART_TransparentTimeout` milliseconds.
 - If the configured ETX characters (`UART_TransparentETX`) have been received on the UART and this trigger has been enabled (see `UART_TransparentETXConfig`).
- To enable a fast transmission of data packets a large MPS of the Bluetooth® LE connection is helpful. The Proteus-e supports up to 243 Bytes Bluetooth® LE packet payload (corresponding to a MTU of 247 Bytes), which may be negotiated by the central device (using a MTU request). If no MTU request is requested by the connecting central device the value of 19 Bytes payload per Bluetooth® LE packet and connection interval as given by the Bluetooth® 4.0 standard is used (compatibility mode to Bluetooth® LE 4.0 devices).
 - The pin *BUSY* shall be used as a kind of flow control for the data transmission in transparent mode. By default the pin level is LOW. As soon as the above mentioned trigger has been detected, the pin switches to HIGH and data transmission starts via Bluetooth® LE. The pin switches to LOW again, as soon as Bluetooth® LE data transmission has finished and the transmission of new data is feasible again. In case the pin is HIGH, no more data is accepted on the UART.

Figure 15: Switch of the *BUSY* pin when transmitting data

To use the signal on the *BUSY* pin as flow control on the host controller side, we recommend to use an OR gate to combine the */RTS* and *BUSY* pins' signals.

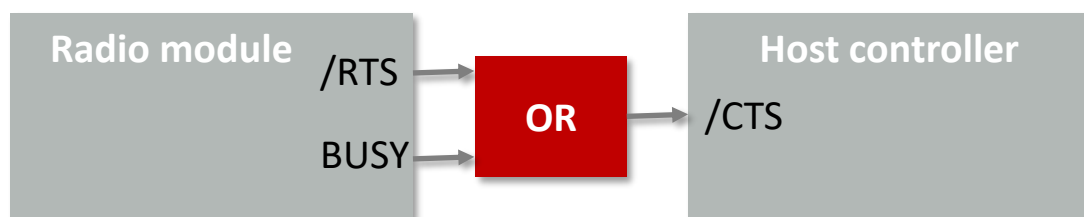
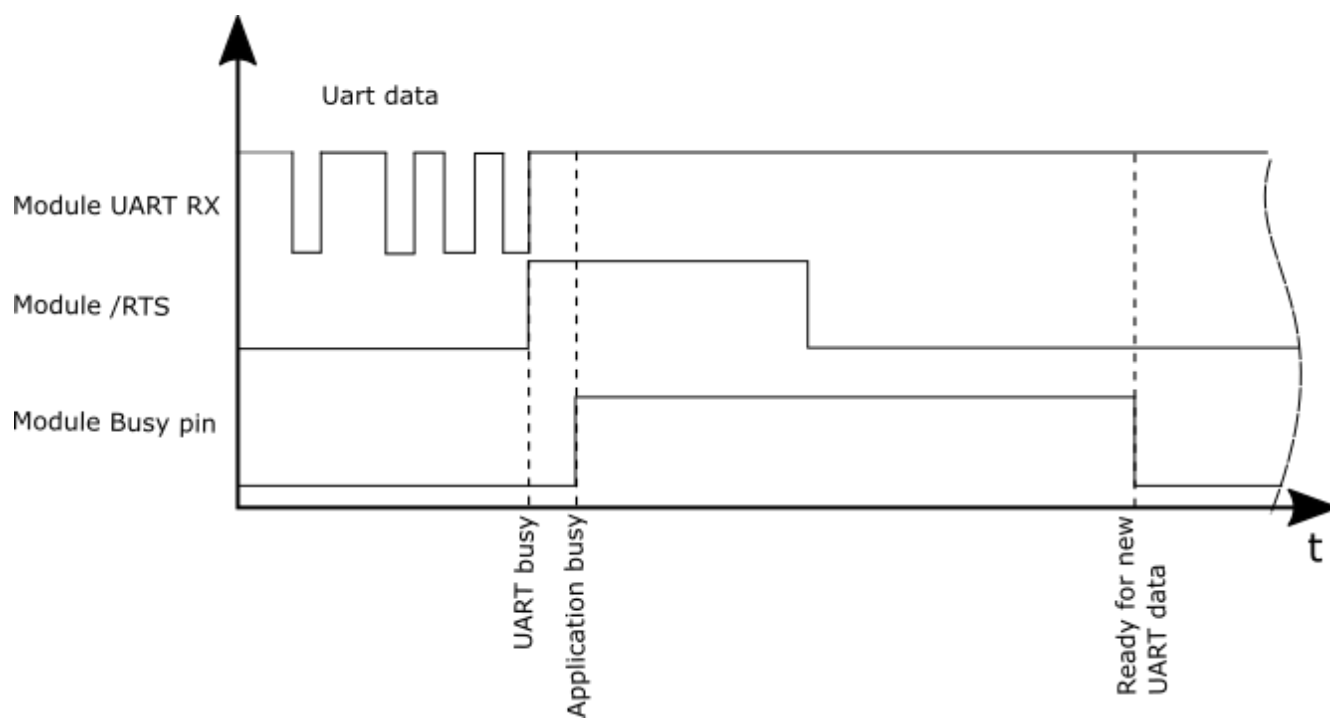


Figure 16: Handling the /RTS and BUSY pin

13. Remote GPIO control

The Proteus-e allows to control free GPIOs via remote access. Chapter 9.6 contains the description of the necessary commands.

To use the remote GPIO control feature of the Proteus-e, the GPIOs of interest must be configured first. This can be done in two ways. Either by the local host (see figure 17), when the radio module is in idle mode (not connected via Bluetooth® LE), or via the connected remote device (see figure 18).

In case of the local host, it must send a `CMD_GPIO_LOCAL_WRITECONFIG_REQ` command to the radio module via UART. In case of the remote device, it must setup a Bluetooth® LE connection to the remote device first and send a `CMD_GPIO_REMOTE_WRITECONFIG_REQ` command to the radio module via Bluetooth® LE afterwards.

The configuration is stored in flash memory, such that it is retained also after a device restart. It can be reset to default by using the `CMD_FACTORYRESET_REQ` command.

The configuration can be also read out using the respective commands, `CMD_GPIO_LOCAL_READCONFIG_REQ` via local host or `CMD_GPIO_REMOTE_READCONFIG_REQ` via remote device.



For security reasons, the remote configuration of GPIOs can be blocked by setting the corresponding bit in the user setting `CFG_Flags`.

If the configuration has been done, the configured GPIOs can be controlled by the local host controller or by any remote device.

To control a GPIO via local host controller just send the respective commands, `CMD_GPIO_LOCAL_WRITE_REQ` for setting GPIO output values (see figure 21), or `CMD_GPIO_LOCAL_READ_REQ` for reading GPIO values (see figure 22). Each time the GPIOs are written via local host, the connected remote device is informed using a `CMD_GPIO_LOCAL_WRITE_IND` message.

To control a GPIO via remote device, first setup a Bluetooth® LE connection to the radio module and send the respective commands, `CMD_GPIO_REMOTE_WRITE_REQ` for setting GPIO output values (see figure 23), or `CMD_GPIO_REMOTE_READ_REQ` for reading GPIO values (see figure 24). Each time the GPIOs are written via remote connection, the local host is informed using a `CMD_GPIO_REMOTE_WRITE_IND` message.

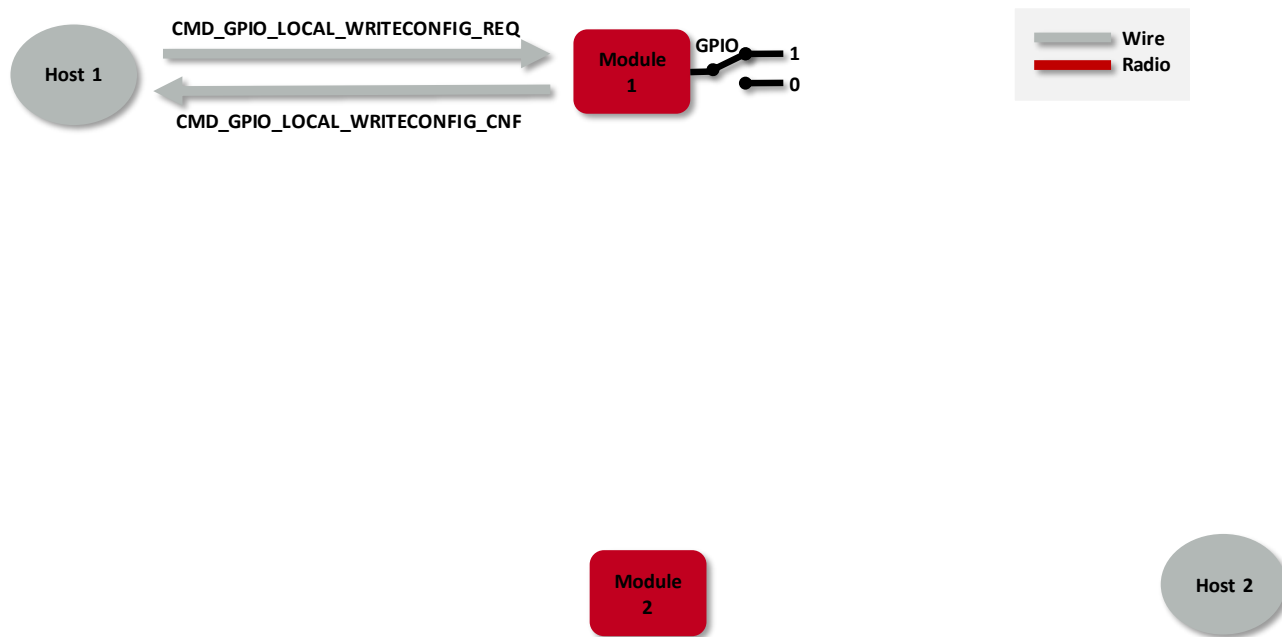


Figure 17: Configure the local GPIOs via local host

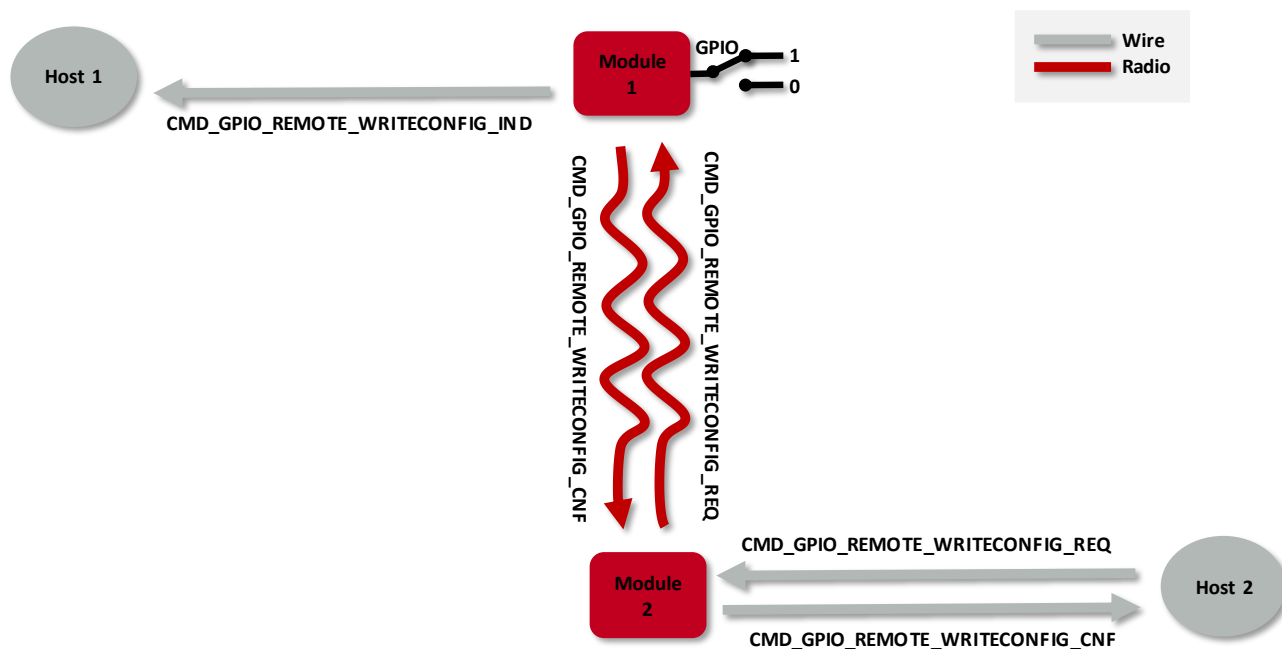


Figure 18: Configure the local GPIOs via remote device host

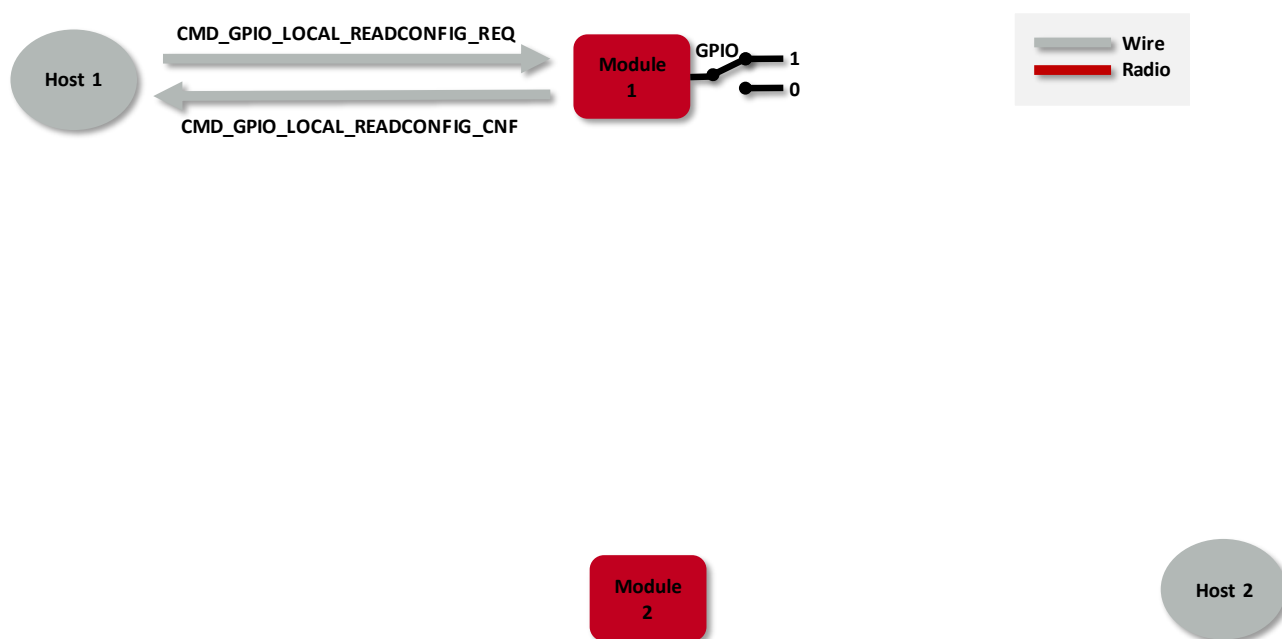


Figure 19: Read the configuration of the local GPIOs via local host

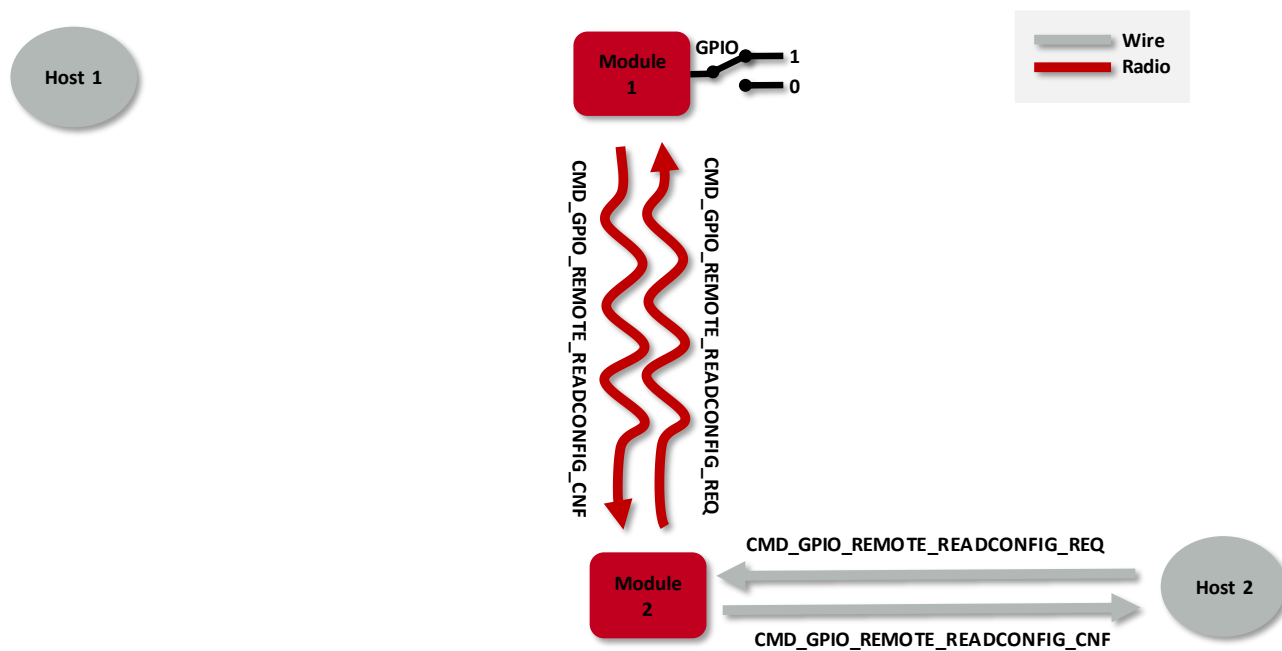


Figure 20: Read the configuration of the local GPIOs via remote device host

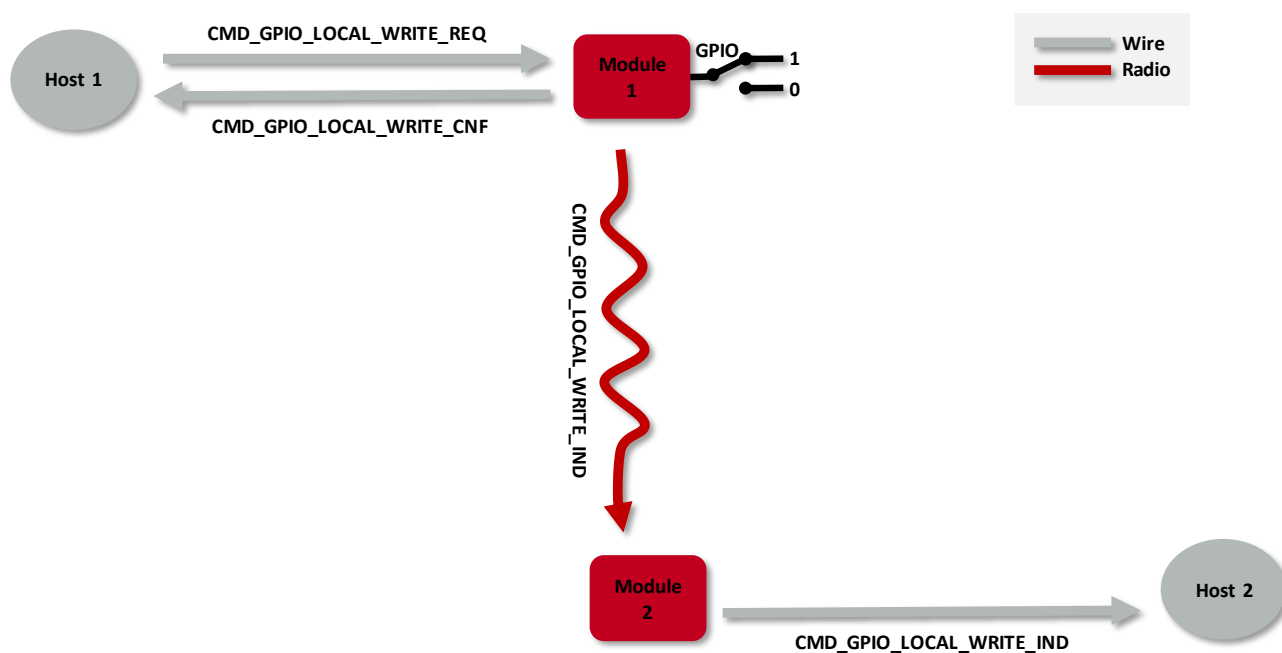


Figure 21: Set the output value of a GPIO via host controller

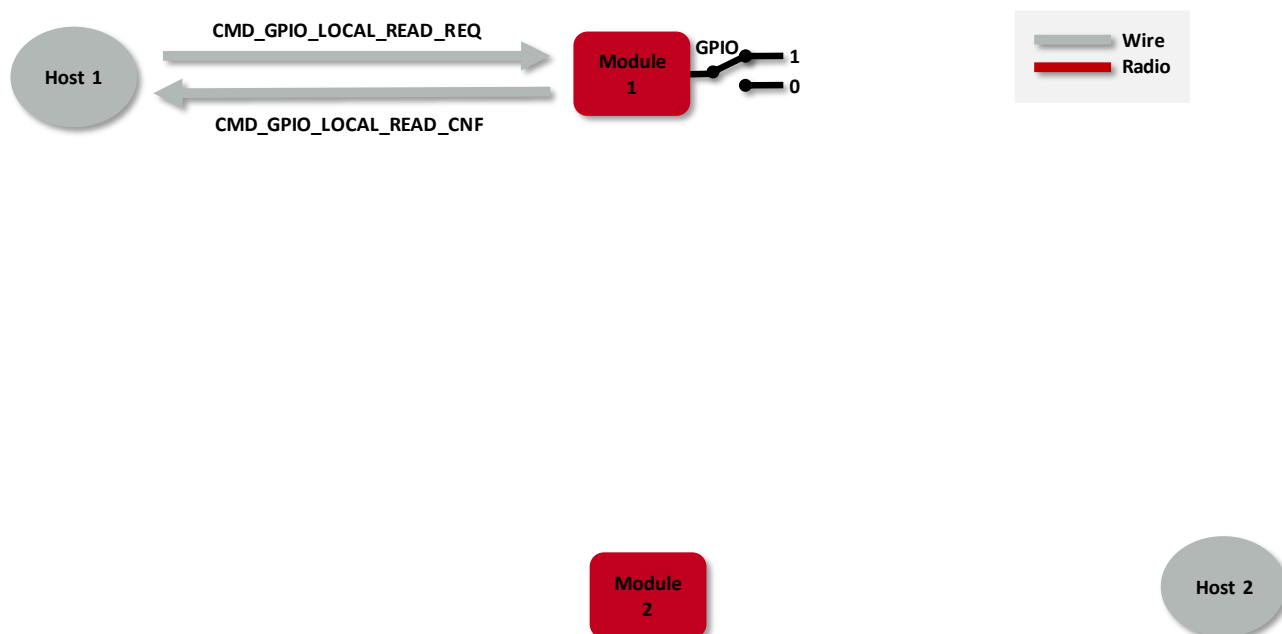


Figure 22: Read the input value of a GPIO via host controller

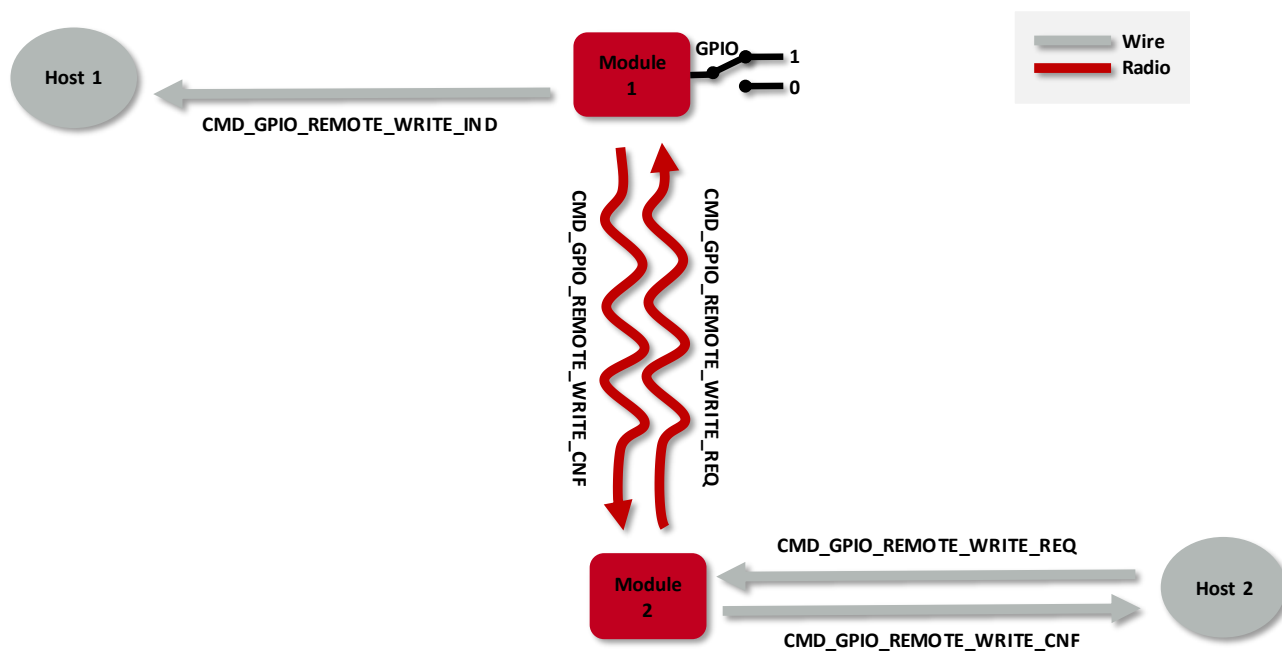


Figure 23: Set the output value of a GPIO via remote device

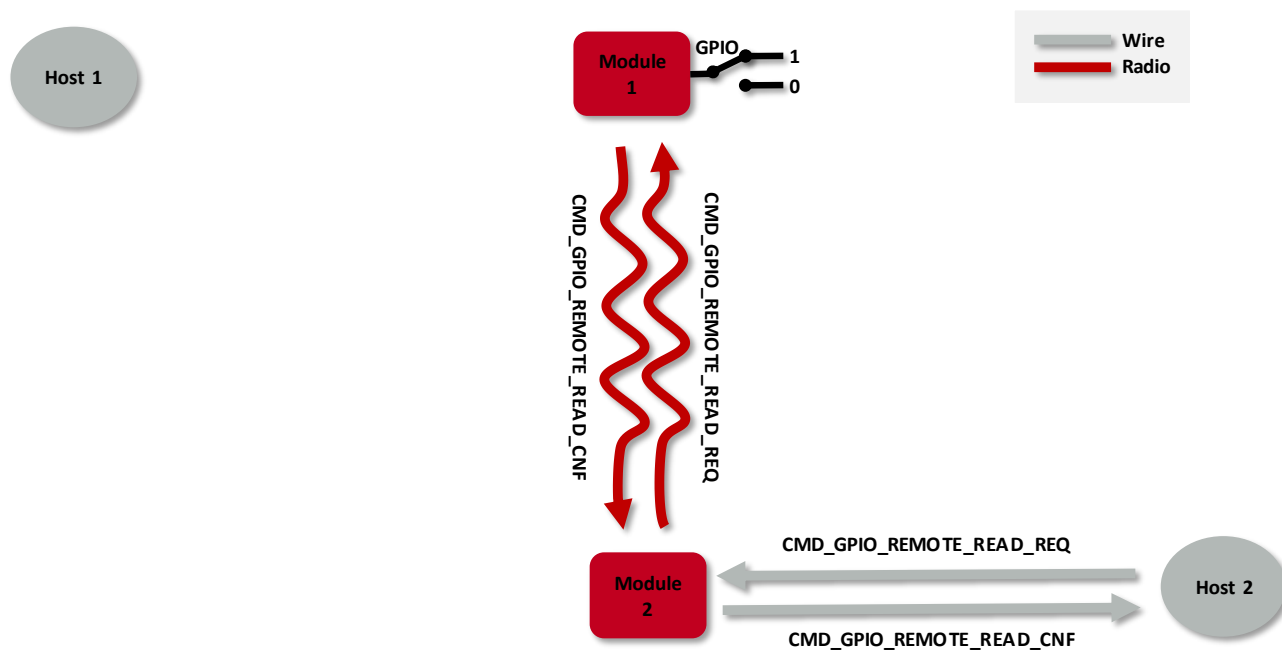
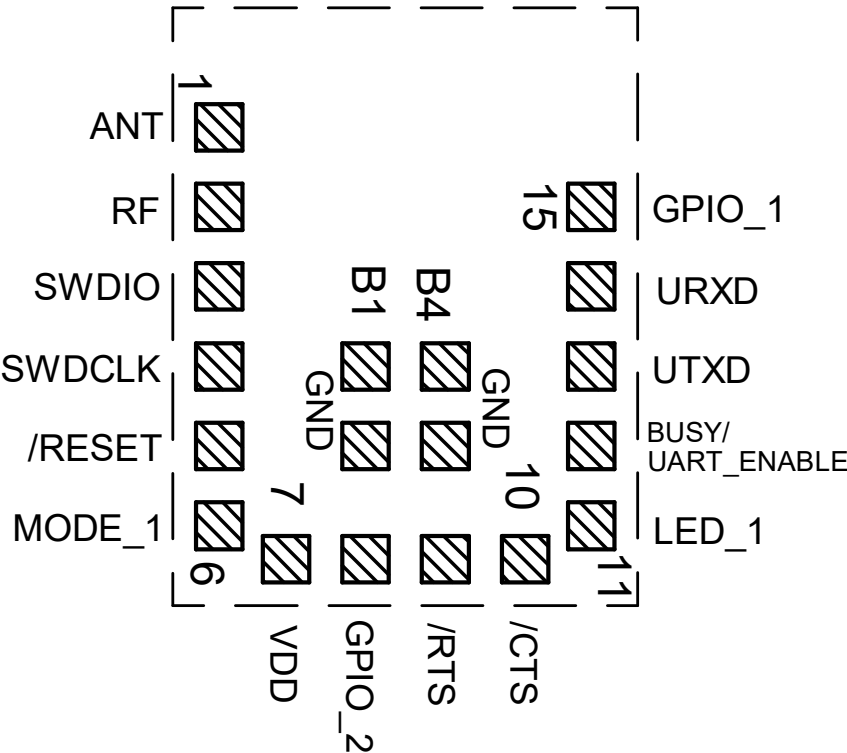


Figure 24: Read the input value of a GPIO via remote device

13.1. Supported GPIO_IDs for remote and local control

The following GPIOs of the Proteus-e are supported for remote and local access.



| Pin | GPIO_ID | Supported functions |
|--------|---------|-------------------------|
| GPIO_1 | 1 | Digital input or output |
| GPIO_2 | 2 | Digital input or output |

Table 25: Supported GPIO_IDs

14. Customizing the Proteus-e

14.1. UUID

The UUID is a unique number identifying a Bluetooth® LE profile and thus describing its functions. The Proteus-e using its standard UUID is compatible to all devices that implement the SPP-like profile, whichever device it is integrated into.

To suspend this interoperability, the user settings RF_SPPBaseUUID, RF_SPPServiceUUID, RF_SPPTXUUID and RF_SPPRXUUID can be used to modify the UUID of the SPP-like profile. With this, a new custom SPP-like profile is defined that is solely known to those that chose the new UUID.

The SPP-like profile consists of the 128 bit base UUID plus the 16 bit UUIDs for the underlying characteristics and services:

| Characteristic | UUID |
|--------------------------|--|
| 128 Bit RF_SPPBaseUUID | 0x6E40xxxx-C352-11E5-953D-0002A5D5C51B |
| 16 Bit RF_SPPServiceUUID | 0x0001 |
| 16 Bit RF_SPPRXUUID | 0x0002 |
| 16 Bit RF_SPPTXUUID | 0x0003 |

Table 26: UUID default values

Using these user settings, the UUIDs of all characteristics calculate as the base UUID, where byte 2 and 3 are replaced by the underlying service or characteristic UUID.

Example:

With the above mentioned default values, the full UUIDs calculate as

| Direction | Characteristic | 128 Bit UUID |
|-----------------------|-------------------|---|
| | Primary service | 0x6E40 0001 -C352-11E5-953D-0002A5D5C51B |
| Remote peer to module | RX characteristic | 0x6E40 0002 -C352-11E5-953D-0002A5D5C51B |
| Module to remote peer | TX characteristic | 0x6E40 0003 -C352-11E5-953D-0002A5D5C51B |

To generate a custom base UUID the Bluetooth® SIG recommends to use the tool:

<http://www.uuidgenerator.net/>

14.2. Appearance

The appearance of the Bluetooth® device is a 2 Bytes value defined by the Bluetooth® SIG. It can be configured by adapting the parameter RF_Appearance.

15. Custom firmware and configuration



Any kind of configuration and firmware, which is provided as Intel hex file, can be programmed on the radio module at Würth Elektronik eiSos production site.

In case of interest, please contact your Business Development Manager (BDM) or WCS@we-online.com.

15.1. Custom configuration of standard firmware

The configuration of the standard firmware includes adoption of the non-volatile settings to customer requirements and creating a customized product based on the standard product.

This variant will result in a customer exclusive module with a unique ordering number. It will also freeze the firmware version to a specific and customer tested version and thus results in a customer exclusive module with a unique ordering number.

Further scheduled firmware updates of the standard firmware will not be applied to this variant automatically. Applying updates or further functions require a customer request and release procedure.

15.2. Customer specific firmware

A customer specific firmware may include "Custom configuration of standard firmware" plus additional options or functions and tasks that are customer specific and not part of the standard firmware.

Further scheduled firmware updates of the standard firmware will not be applied to this variant automatically. Applying updates or further functions require a customer request and release procedure.

This also results in a customer exclusive module with a unique ordering number.

An example for this level of customization are functions like host-less operation where the module will perform data generation (e.g. by reading a SPI or I²C sensor) and cyclic transmission of this data to a data collector, while sleeping or being passive most of the time.

Also replacing UART with SPI as host communication interface is classified as a custom specific option.

Certification critical changes need to be re-evaluated by an external qualified measurement laboratory. These critical changes may occur when e.g. changing radio parameters, the channel access method, the duty-cycle or in case of various other functions and options possibly used or changed by a customer specific firmware.

15.3. Customer firmware

A customer firmware is a firmware written and tested by the customer himself or a 3rd party as a customer representative specifically for the hardware platform provided by a module.

This customer firmware (e.g. in form of an Intel hex file) will be implemented into the module's production process at our production site.

This also results in a customer exclusive module with a unique ordering number. The additional information needed for this type of customer firmware, such as hardware specific details and details towards the development of such firmware are not available for the public and can only be made available to qualified customers.



The qualification(s) and certification(s) of the standard module cannot be applied to this customer firmware solution without a review and verification.

16. Firmware updates

All products will experience maintenance, security and/or feature updates from time to time. For the standard products these are maintained via the PCN process.

Customers can request the creation of a customized product including a "firmware freeze" to ensure that they will receive their verified product even if the standard product is updated.



The Proteus-e does not support any kind of update of the production firmware images by users.

16.1. Firmware flashing using the production interface

The Proteus-e offers a serial wire debug and programming interface (SWD) for module flash access. This interface can be used by customers to unlock and erase the entire chip and install their own firmware.

This interface is not intended to perform updates of Proteus-e firmware by users.

Production firmware images and binary files for Würth Elektronik eiSos wireless connectivity modules are not publicly available in general.



Customers performing any flashing of own firmware are fully responsible for any certification, declaration listing and qualification as the Proteus-e documentation becomes invalid.

17. Firmware history

Version 1.0.0 "Release"

- Initial version

Version 1.1.0 "Release"

- Added option to set the FS_BTMAC
- Added new options for improved data transmission in transparent mode (UART_TransparentTimeout, UART_TransparentMaxPayload, UART_TransparentETX, UART_TransparentETXConfig)
Please find more information in chapter `Transparent mode`.
- Added option to request the maximum transmission unit (MTU) via remote command. For more details, please refer to the Proteus-e advanced developer guide [2]

17.1. Known issues

| Index | Details | Affected versions |
|--------|--|-------------------|
| KI-001 | <p>Description: Setting the Bluetooth® reserved UUID 0000xxxx-0000-1000-8000-00805f9b34fb as base UUID will result in malfunctioning. The module will not start-up again.</p> <p>Affected functions: User setting RF_SPPBaseUUID and command CMD_SET_REQ.</p> <p>Workaround: Do not use CMD_SET_REQ function to set the user setting RF_SPPBaseUUID to the value 0000xxxx-0000-1000-8000-00805f9b34fb .</p> | 1.0.0 |

18. Hardware history

Version 1.0 "Release"

- Initial release

19. Design in guide

19.1. Advice for schematic and layout

For users with less RF experience it is advisable to closely copy the relating EV-Board with respect to schematic and layout, as it is a proven design. The layout should be conducted with particular care, because even small deficiencies could affect the radio performance and its range or even the conformity.

The following general advice should be taken into consideration:

- A clean, stable power supply is strongly recommended. Interference, especially oscillation can severely restrain range and conformity.
- Variations in voltage level should be avoided.
- LDOs, properly designed in, usually deliver a proper regulated voltage.
- Blocking capacitors and a ferrite bead in the power supply line can be included to filter and smoothen the supply voltage when necessary.
- Elements for ESD protection should be placed on all pins that are accessible from the outside and should be placed close to the accessible area. For example, the RF-pin is accessible when using an external antenna and should be protected.
- ESD protection for the antenna connection must be chosen such as to have a minimum effect on the RF signal. For example, a protection diode with low capacitance such as the 8231606A or a 68 nH air-core coil connecting the RF-line to ground give good results.
- Placeholders for optional antenna matching or additional filtering are recommended.
- The antenna path should be kept as short as possible.
- The use of an external reset IC should be considered if one of the following points is relevant:
 - The slew rate of the power supply exceeds the electrical specifications.
 - The effect of different current consumptions on the voltage level of batteries or voltage regulators should be considered. The module draws higher currents in certain scenarios like start-up or radio transmit which may lead to a voltage drop on the supply. A restart under such circumstances should be prevented by ensuring that the supply voltage does not drop below the minimum specifications.
 - Voltage levels below the minimum recommended voltage level may lead to malfunction. The reset pin of the module shall be held on LOW logic level whenever the VDD is not stable or below the minimum operating Voltage.
 - Special care must be taken in case of battery powered systems.
- To avoid the risk of short circuits and interference there should be no routing underneath the module on the top layer of the baseboard.
- On the second layer, a ground plane is recommended, to provide good grounding and shielding to any following layers and application environment.

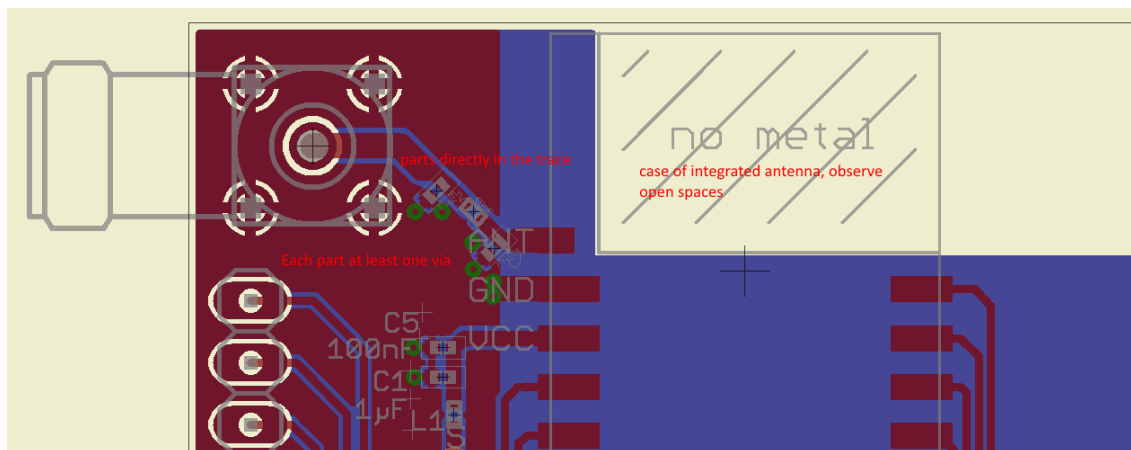


Figure 25: Layout

- In case of integrated antennas it is required to have areas free from ground. This area should be copied from the EV-Board.
- The area with the integrated antenna must overlap with the carrier board and should not protrude, as it is matched to sitting directly on top of a PCB.
- Modules with integrated antennas should be placed with the antenna at the edge of the main board. It should not be placed in the middle of the main board or far away from the edge. This is to avoid tracks beside the antenna.
- Filter and blocking capacitors should be placed directly in the tracks without stubs, to achieve the best effect.
- Antenna matching elements should be placed close to the antenna / connector, blocking capacitors close to the module.
- Ground connections for the module and the capacitors should be kept as short as possible and with at least one separate through hole connection to the ground layer.
- ESD protection elements should be placed as close as possible to the exposed areas.



Fixed values can not be recommended, as these depend on the circumstances of the application (main power source, interferences etc.).

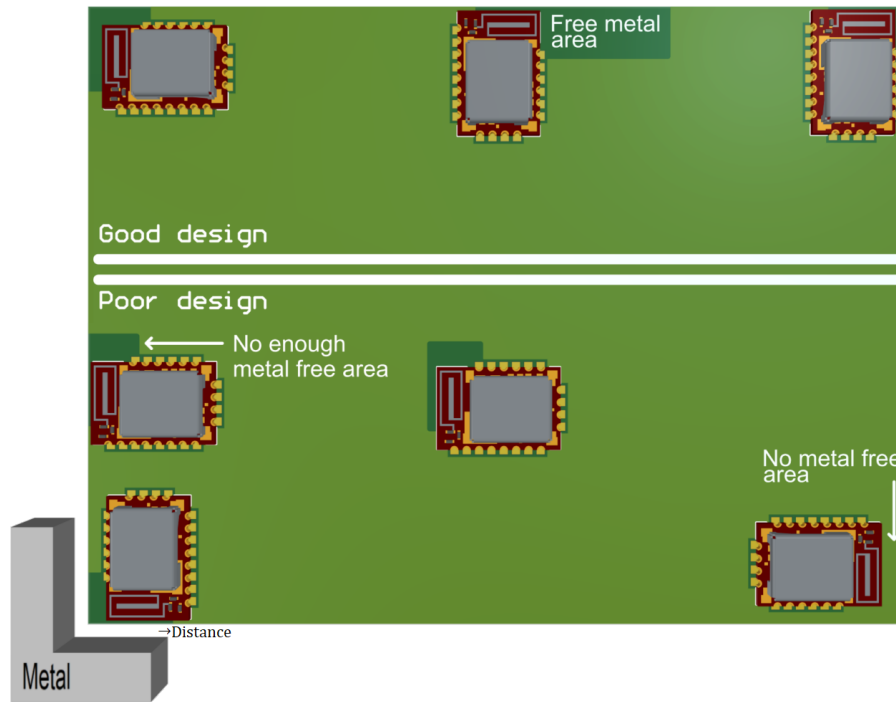


Figure 26: Placement of the module with integrated antenna

19.2. Designing the antenna connection

The antenna should be connected with a $50\ \Omega$ line. This is needed to obtain impedance matching to the module and avoids reflections. Here we show as an example how to calculate the dimensions of a $50\ \Omega$ line in form of a micro strip above ground, as this is easiest to calculate. Other connections like coplanar or strip line are more complicated to calculate but can offer more robustness to EMC. There are free calculation tools available in the internet.

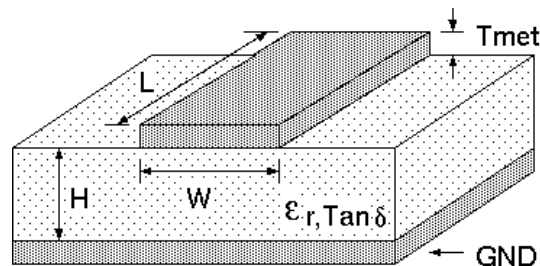


Figure 27: Dimensioning the antenna connection as micro strip

The width W for a micro strip can be calculated using the following equation:

$$W = 1.25 \times \left(\frac{5.98 \times H}{e^{\frac{50 \times \sqrt{\epsilon_r + 1.41}}{87}}} - T_{met} \right)$$

Example:

A FR4 material with $\epsilon_r = 4.3$, a height $H = 1000\ \mu\text{m}$ and a copper thickness of $T_{met} = 18\ \mu\text{m}$ will lead to a trace width of $W \sim 1.9\ \text{mm}$. To ease the calculation of the micro strip line (or e.g. a

coplanar) many calculators can be found in the internet.

- As rule of thumb a distance of about $3 \times W$ should be observed between the micro strip and other traces / ground.
- The micro strip refers to ground, therefore there has to be the ground plane underneath the trace.
- Keep the feeding line as short as possible.

19.3. Antenna solutions

There exist several kinds of antennas, which are optimized for different needs. Chip antennas are optimized for minimal size requirements but at the expense of range, PCB antennas are optimized for minimal costs, and are generally a compromise between size and range. Both usually fit inside a housing.

Range optimization in general is at the expense of space. Antennas that are bigger in size, so that they would probably not fit in a small housing, are usually equipped with a RF connector. A benefit of this connector may be to use it to lead the RF signal through a metal plate (e.g. metal housing, cabinet).

As a rule of thumb a minimum distance of $\lambda / 10$ (which is 3.5 cm @ 868 MHz and 1.2 cm @ 2.44 GHz) from the antenna to any other metal should be kept. Metal placed further away will not directly influence the behavior of the antenna, but will anyway produce shadowing.



Keep the antenna as far as possible from large metal objects to avoid electromagnetic field blocking.

In the following chapters, some special types of antenna are described.

19.3.1. Wire antenna

An effective antenna is a $\lambda / 4$ radiator with a suiting ground plane. The simplest realization is a piece of wire. It's length is depending on the used radio frequency, so for example 8.6 cm 868.0 MHz and 3.1 cm for 2.440 GHz as frequency. This radiator needs a ground plane at its feeding point. Ideally, it is placed vertically in the middle of the ground plane. As this is often not possible because of space requirements, a suitable compromise is to bend the wire away from the PCB respective to the ground plane. The $\lambda / 4$ radiator has approximately 40 Ω input impedance. Therefore, matching is not required.

19.3.2. Chip antenna

There are many chip antennas from various manufacturers. The benefit of a chip antenna is obviously the minimal space required and reasonable costs. However, this is often at the expense of range. For the chip antennas, reference designs should be followed as closely as possible, because only in this constellation can the stated performance be achieved.

19.3.3. PCB antenna

PCB antenna designs can be very different. The special attention can be on the miniaturization or on the performance. The benefits of the PCB antenna are their small / not existing (if PCB space is available) costs, however the EV of a PCB antenna holds more risk of failure than the use of a finished antenna. Most PCB antenna designs are a compromise of range and space between chip antennas and connector antennas.

19.3.4. Antennas provided by Würth Elektronik eiSos

Besides the radio modules Würth Elektronik eiSos provides various antennas tailored for the different frequency bands. The recommended single external antennas are shown in the subsequent chapters.



In case integrated multilayer chip antennas are needed because of space limitations, please refer to
<https://www.we-online.com/en/components/products/WE-MCA>.

19.3.4.1. 2600130021 - Himalia dipole antenna



Figure 28: Himalia dipole antenna

Due to the fact that the antenna has dipole topology, there is no need for an additional ground plane. Nevertheless, the specification was measured edge mounted and 90 ° bent on a 100 x 100 mm ground plane.

| Specification | Value |
|--|------------------|
| Frequency range [GHz] | 2.4 – 2.5 |
| Impedance [Ω] | 50 |
| VSWR | $\leq 2:1$ |
| Polarization | Linear |
| Radiation | Omni-Directional |
| Peak Gain [dBi] | 2.8 |
| Average Gain [dBi] | -0.6 |
| Efficiency | 85 % |
| Dimensions (L x d) [mm] | 83.1 x 10 |
| Weight [g] | 7.4 |
| Connector | SMA plug |
| Operating temp. [$^{\circ}\text{C}$] | -40 – +80 |

Special care must be taken for FCC certification when using this external antenna to fulfill the requirement of permanently attached antenna or unique coupling, for example by using the certified dipole antenna in a closed housing, so that it is possible to remove it only through professional installation.

20. Reference design

Proteus-e was tested and certified on the corresponding Proteus-e EV-Board. For the European Conformity the EV-Board serves as reference design. When reusing Würth Elektronik eiSos FCC or IC certification it is mandatory to follow the trace design.

Complete layout and schematic information can be found in the manual of the Proteus-e EV-Board.



The reference design is the same for the Proteus-e and Ophelia-I radio modules. The module's pins on the schematic are named according to Ophelia-I module. The pin compatibility table between Ophelia-I and Proteus-e is shown in the schematic.

20.1. EV-Board

20.1.1. Schematic

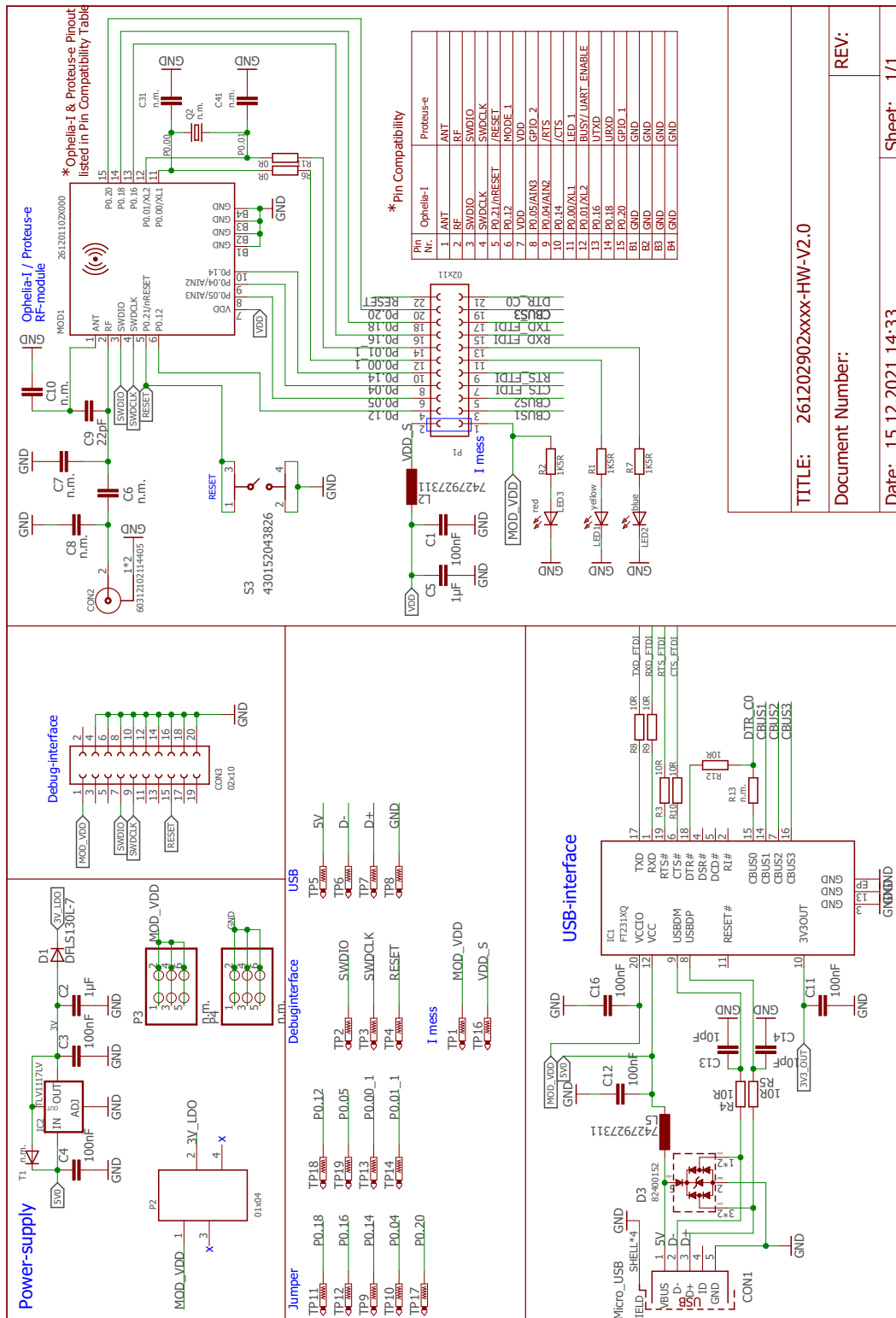


Figure 29: Reference design: Schematic

20.2. Layout

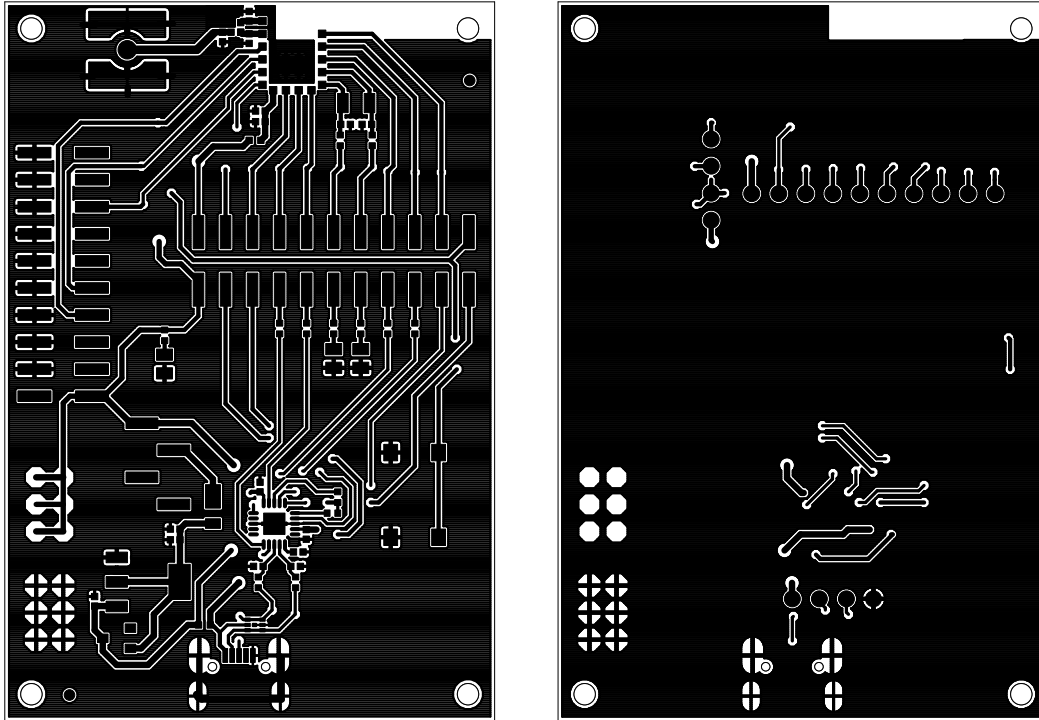


Figure 30: Top layer (top), bottom layer (bottom)

20.3. Internal antenna radiation characteristics

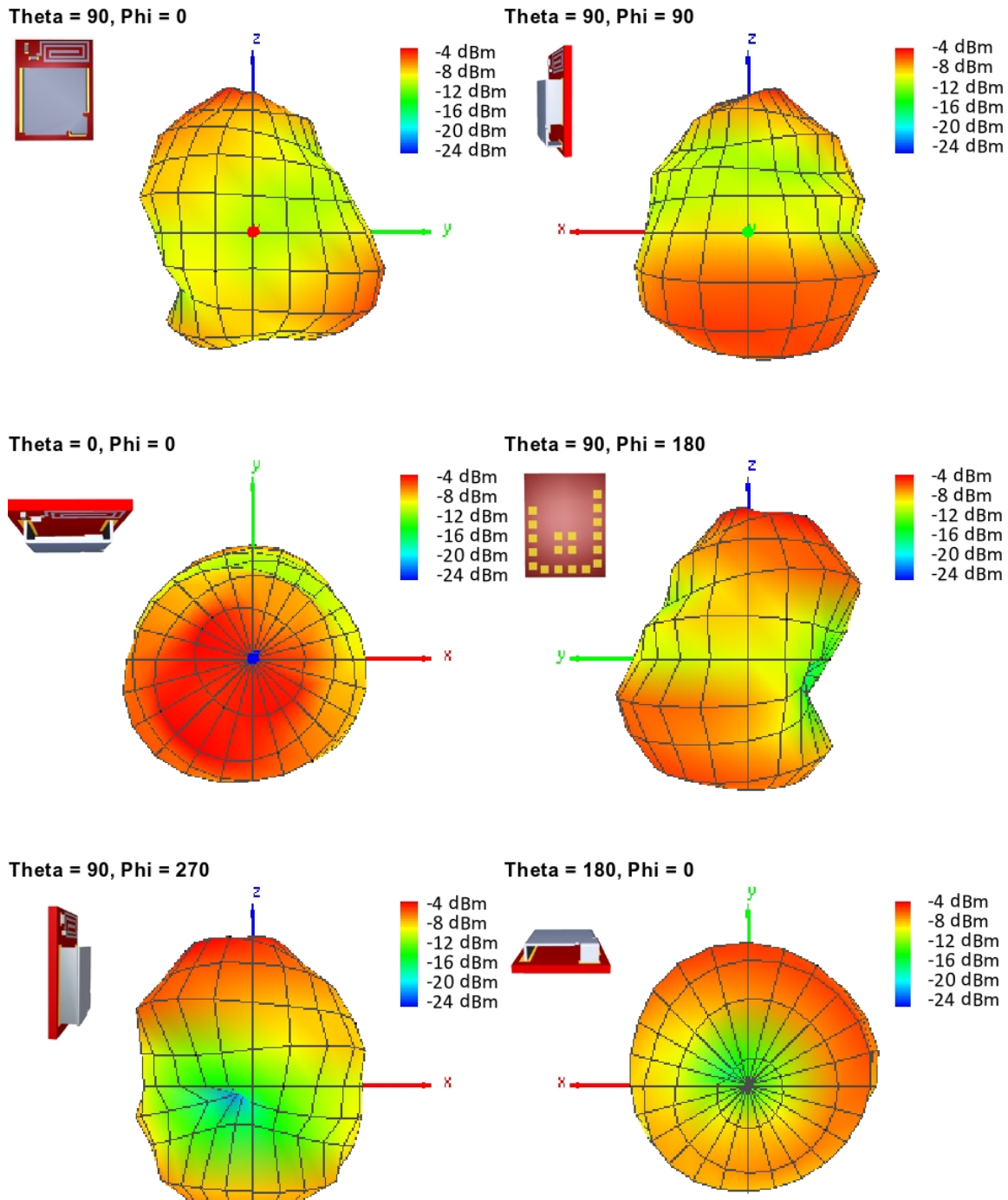


Figure 31: Antenna characteristic from integrated antenna measured on official EV-Board¹

¹Radiation characteristic shown is valid for the module on the EV-Board. It is important to be aware that size of groundplane and placement of module has influence on the radiation pattern.

20.4. Trace design

The trace design shown in this is proved for FCC and IC certification.



To reuse and reference to the Würth Elektronik eiSos' FCC ID it is mandatory to use the trace design.

To provide best options following trace designs are certified:

- A simple short between the pins *RF* and *ANT* pin to be used if size and price is critical and the range is uncritical. See chapter 20.4.1
- A 22 pF capacitor connecting *RF* and *ANT* pin to be used if size and price is less critical, but an assembly variant with external antenna is also being used. See chapter 20.4.2
- A 22 pF capacitor connecting *RF* pin to the external pre-certified dipole antenna (Himalia [11] , chapter 19.3.4.1). This configuration suits best if size and price is less critical, but radio range should be optimized. See chapter 20.4.3

The trace designs use the same layer stack up.

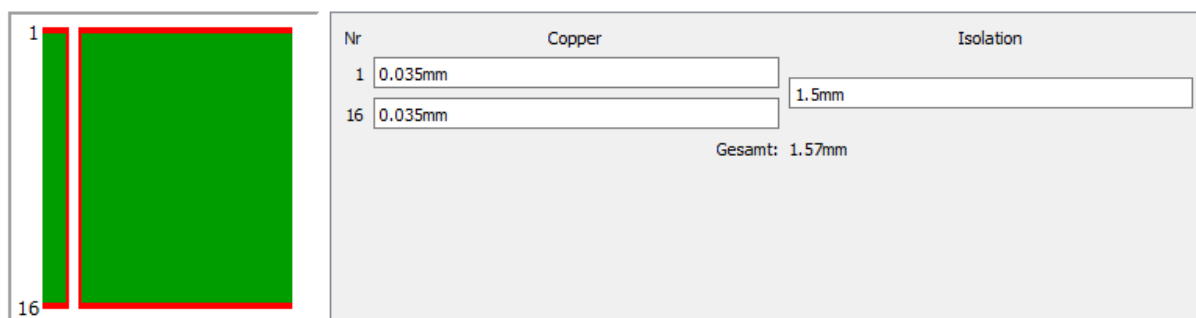


Figure 32: Stack-up

- Top layer is used for routing, filled with ground plane except the area under the module and the antenna free area.
- Bottom layer is the ground plane with as few as possible routing dividing it.



In the following the light green marked areas are the trace designs to be followed when reusing certifications.

20.4.1. Simple short using internal antenna

The simple short is a 50 Ω coplanar strip connecting *RF* and *ANT* pin. Figures 33 and 34 show this in detail.

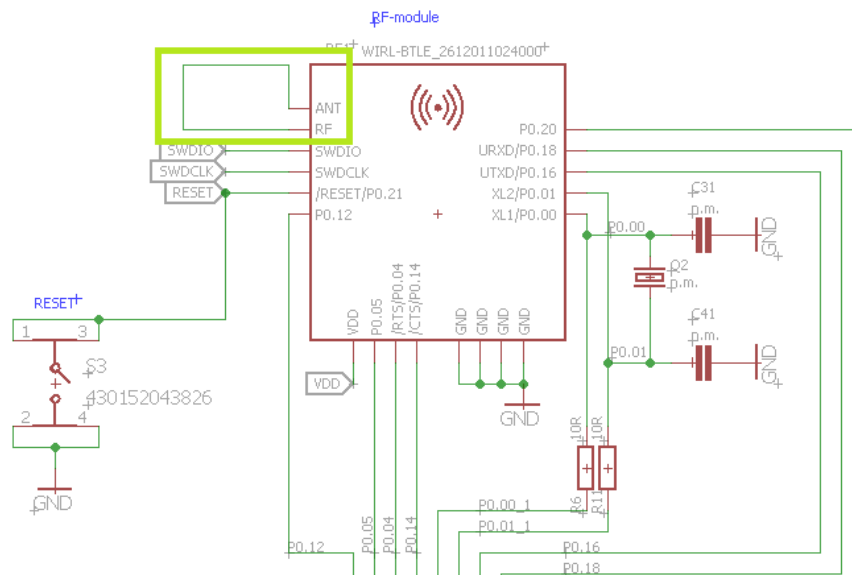


Figure 33: Simple short schematic

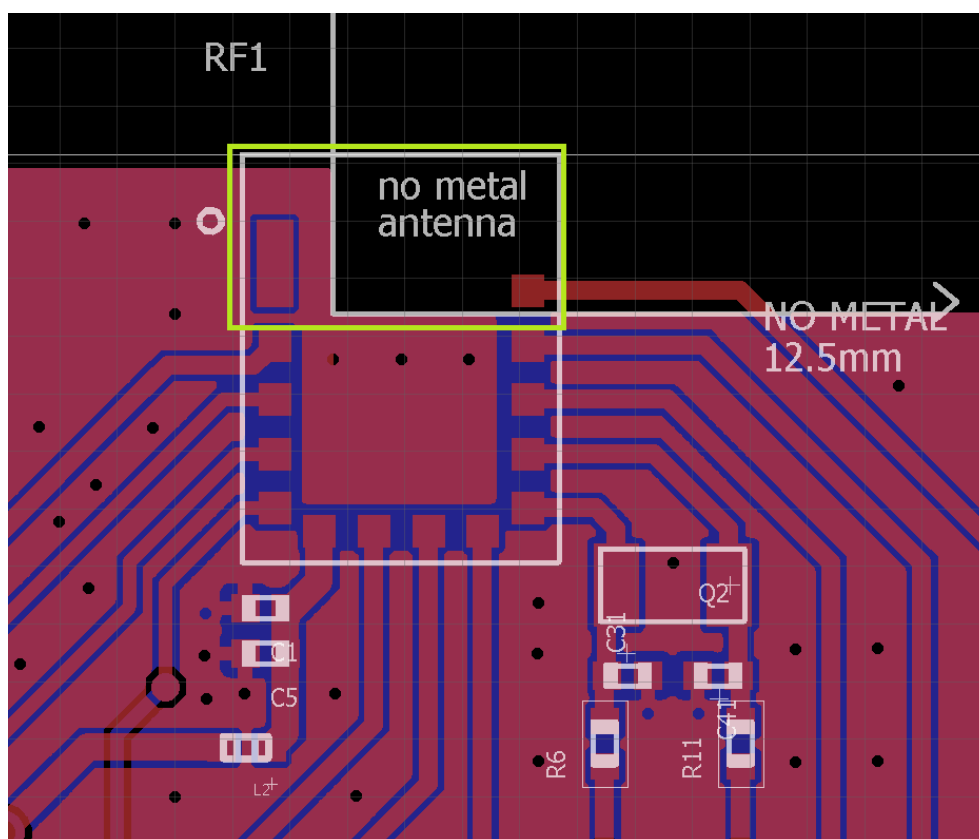


Figure 34: Simple short layout

20.4.2. 22 pF coupling capacitor using internal antenna

In this configuration, instead of the simple short, a 22 pF capacitor is used at C9 connecting *RF* and *ANT* pin. C6, C7, C8 and C10 are left unassembled. Figures 35 and 36 show this in detail.

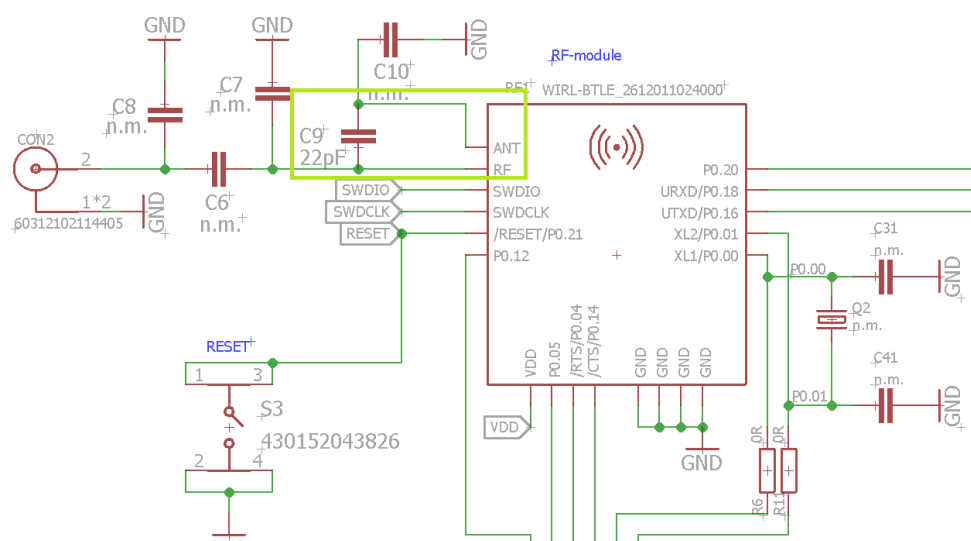


Figure 35: Capacitor internal antenna schematic

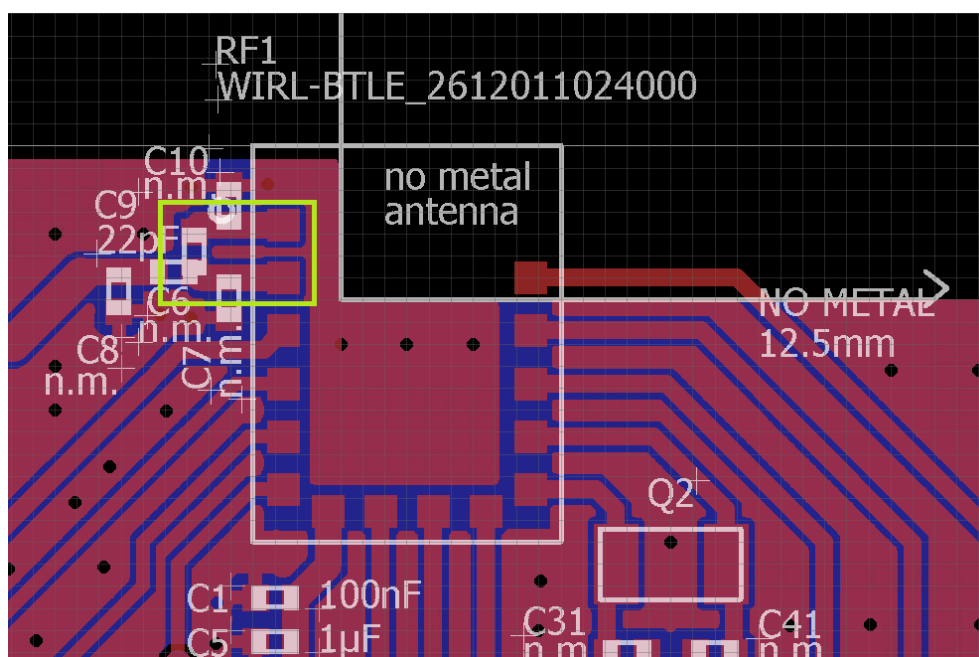


Figure 36: Capacitor internal antenna layout

20.4.3. 22 pF coupling capacitor using external antenna

In this configuration, the 22 pF capacitor is used at C6 connecting the *RF* pin to a dipol antenna (Himalia [11], chapter 19.3.4.1). C7, C8, C9 and C10 are left unassembled. Figures 37 and 38 show this in detail.

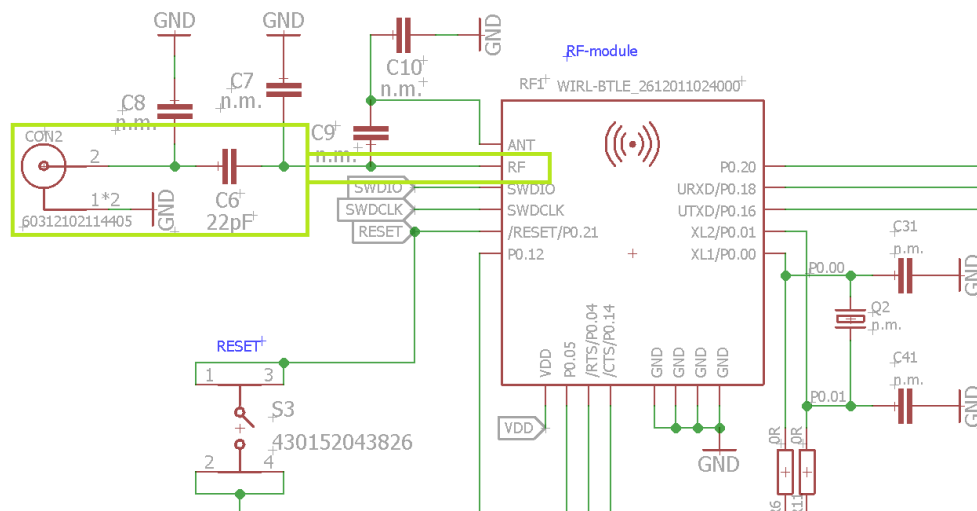


Figure 37: Capacitor external antenna schematic

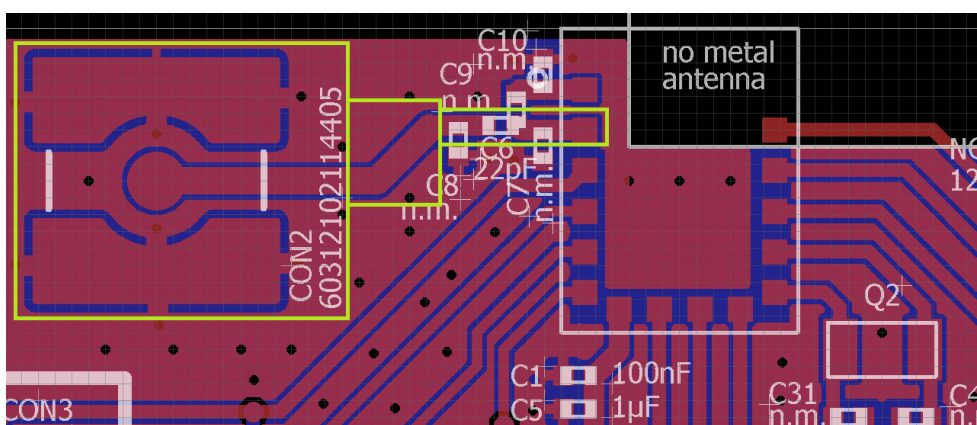


Figure 38: Capacitor external antenna layout



To fulfill §15.203 of FCC, the manufacturer of the end device must ensure that no antenna other than that furnished by the responsible party shall be used with the device. The use of a permanently attached antenna or an antenna, that uses a unique coupling to the end device, shall be considered sufficient to comply.

20.5. Antenna fine tuning

Engineers with experience in radio design and the needed measurement equipment should consider the possibility of antenna tuning. The smart antenna connection provides the possibility to even tune the antenna inside the module.

Due to the influence of mounting conditions as metallic objects close to the antenna, or the size of the mother-pcb and ground plane there might be some detuning of the antenna. Adjusting might be done by measuring the resulting antenna impedance and using corresponding values for C7, C9 and C10 see Figure 35.

This fine tuning is violating the trace design in the sense of FCC rules. A radio spot check measurement for the end device is needed.

Using the trace design option 20.4.2 as implemented on the EV-Board gives the possibility to either follow the trace design or do fine tuning if needed without changing the PCB design.

21. Manufacturing information

21.1. Moisture sensitivity level

This wireless connectivity product is categorized as JEDEC Moisture Sensitivity Level 3 (MSL3), which requires special handling.

More information regarding the MSL requirements can be found in the IPC/JEDEC J-STD-020 standard on www.jedec.org.

More information about the handling, picking, shipping and the usage of moisture/reflow and/or process sensitive products can be found in the IPC/JEDEC J-STD-033 standard on www.jedec.org.

21.2. Soldering

21.2.1. Reflow soldering

Attention must be paid on the thickness of the solder resist between the host PCB top side and the modules bottom side. Only lead-free assembly is recommended according to JEDEC J-STD020.

| Profile feature | | Value |
|--|---------------------|-------------|
| Preheat temperature, min | $T_{S \text{ Min}}$ | 150 °C |
| Preheat temperature, max | $T_{S \text{ Max}}$ | 200 °C |
| Preheat time from $T_{S \text{ Min}}$ to $T_{S \text{ Max}}$ | t_S | 60 - 120 s |
| Ramp-up rate (T_L to T_P) | | 3 °C/s max. |
| Liquidous temperature | T_L | 217 °C |
| Time t_L maintained above T_L | t_L | 60 - 150 s |
| Peak package body temperature | T_P | 260 °C |
| Time within 5 °C of actual peak temperature | t_P | 20 - 30 s |
| Ramp-down rate (T_P to T_L) | | 6 °C/s max. |
| Time 20 °C to T_P | | 8 min max. |

Table 28: Classification reflow soldering profile, Note: refer to IPC/JEDEC J-STD-020E

It is recommended to solder this module on the last reflow cycle of the PCB. For solder paste use a LFM-48W or Indium based SAC 305 alloy (Sn 96.5 / Ag 3.0 / Cu 0.5 / Indium 8.9HF / Type 3 / 89 %) type 3 or higher.

The reflow profile must be adjusted based on the thermal mass of the entire populated PCB, heat transfer efficiency of the reflow oven and the specific type of solder paste used. Based on the specific process and PCB layout the optimal soldering profile must be adjusted and verified. Other soldering methods (e.g. vapor phase) have not been verified and have to be validated by the customer at their own risk. Rework is not recommended.

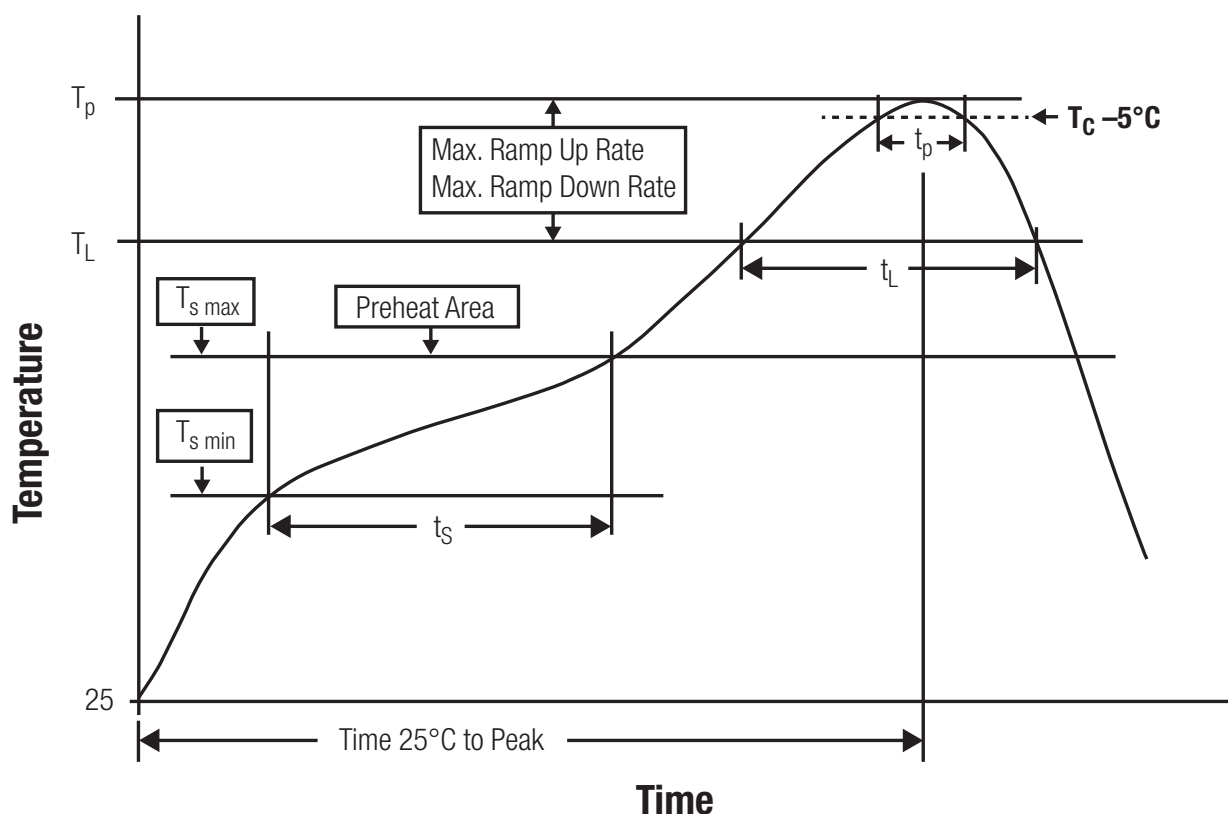


Figure 39: Reflow soldering profile

After reflow soldering, visually inspect the board to confirm proper alignment.

21.2.2. Cleaning

Do not clean the product. Any residue cannot be easily removed by washing. Use a "no clean" soldering paste and do not clean the board after soldering.

- Do not clean the product with water. Capillary effects can draw water into the gap between the host PCB and the module, absorbing water underneath it. If water is trapped inside, it may short-circuit adjoining pads. The water may also destroy the label and ink-jet printed text on it.
- Cleaning processes using alcohol or other organic solvents may draw solder flux residues into the housing, which won't be detected in a post-wash inspection. The solvent may also destroy the label and ink-jet printed text on it.
- Do not use ultrasonic cleaning as it will permanently damage the part, particularly the crystal oscillators.

21.2.3. Potting and coating

- If the product is potted in the customer application, the potting material might shrink or expand during and after hardening. Shrinking could lead to an incomplete seal, allowing contaminants into the component. Expansion could damage components. We recommend a manual inspection after potting to avoid these effects.
- Conformal coating or potting results in loss of warranty.
- The RF shield will not protect the part from low-viscosity coatings and potting. An undefined amount of coating and potting will enter inside the shielding.
- Conformal coating and potting will influence the parts of the radio front end and consequently influence the radio performance.
- Potting will influence the temperature behavior of the device. This might be critical for components with high power.

21.2.4. Other notations

- Do not attempt to improve the grounding by forming metal strips directly to the EMI covers or soldering on ground cables, as it may damage the part and will void the warranty.
- Always solder every pad to the host PCB even if some are unused, to improve the mechanical strength of the module.
- The part is sensitive to ultrasonic waves, as such do not use ultrasonic cleaning, welding or other processing. Any ultrasonic processing will void the warranty.

21.3. ESD handling

This product is highly sensitive to electrostatic discharge (ESD). As such, always use proper ESD precautions when handling. Make sure to handle the part properly throughout all stages of production, including on the host PCB where the module is installed. For ESD ratings, refer to the module series' maximum ESD section. For more information, refer to the relevant chapter 4. Failing to follow the aforementioned recommendations can result in severe damage to the part.

- The first contact point when handling the PCB is always between the local GND and the host PCB GND, unless there is a galvanic coupling between the local GND (for example work table) and the host PCB GND.
- Before assembling an antenna patch, connect the grounds.
- While handling the RF pin, avoid contact with any charged capacitors and be careful when contacting any materials that can develop charges (for example coaxial cable with around 50-80 pF/m, patch antenna with around 10 pF, soldering iron etc.)
- Do not touch any exposed area of the antenna to avoid electrostatic discharge. Do not let the antenna area be touched in a non ESD-safe manner.
- When soldering, use an ESD-safe soldering iron.

21.4. Safety recommendations

It is your duty to ensure that the product is allowed to be used in the destination country and within the required environment. Usage of the product can be dangerous and must be tested and verified by the end user. Be especially careful of:

- Use in areas with risk of explosion (for example oil refineries, gas stations).
- Use in areas such as airports, aircraft, hospitals, etc., where the product may interfere with other electronic components.

It is the customer's responsibility to ensure compliance with all applicable legal, regulatory and safety-related requirements as well as applicable environmental regulations. Disassembling the product is not allowed. Evidence of tampering will void the warranty.

- Compliance with the instructions in the product manual is recommended for correct product set-up.
- The product must be provided with a consolidated voltage source. The wiring must meet all applicable fire and security prevention standards.
- Handle with care. Avoid touching the pins as there could be ESD damage.

Be careful when working with any external components. When in doubt consult the technical documentation and relevant standards. Always use an antenna with the proper characteristics.



Würth Elektronik eiSos radio modules with high output power of up to 500 mW generate a large amount of heat while transmitting. The manufacturer of the end device must take care of potentially necessary actions for his application.

22. Product testing

22.1. Würth Elektronik eiSos in-house production tests

To achieve a high quality standard, Würth Elektronik eiSos follows a philosophy of supplying fully tested radio modules. At the end of the production process, every unit undergoes an optical inspection. Here the quality of soldering, edge castellation and edge milling is monitored.

If this has been passed, the radio modules are handed over to the automatic test equipment for the electrical characterization. This includes:

- Voltage and current tests to ensure proper electrical performance
- RF characteristics (frequency, spectrum, TX power) measurement and calibration
- Radio communication tests
- Firmware and serial number programming
- Host interface communication tests

The automated testing process is logged for internal quality control. The gained measurement data of each unit is analysed to detect defective parts and investigate the corresponding root cause. Defective radio modules are discarded, in order to guarantee a 100% failure-free delivery to customers.

22.2. EMS production tests

The rigorous in-series production testing ensures that EMS don't need to duplicate firmware tests or measurements. This streamlines the process and eliminates the need for additional testing over analogue and digital interfaces during device production. When it comes to device testing, the ideal focus should be on module assembly quality:

- All module pins are soldered properly on the base PCB
- There are no short circuits
- The mounting process did not damage the module
- The communication between host and radio module is working
- The antenna is connected properly

Simple "Go/No go" tests, like checking the RSSI value, give already a hint if the power supply and antenna have been connected properly.

In addition to such standard testing procedures, radio module integrators have the flexibility to perform additional dedicated tests to thoroughly evaluate the device. Specific tests they can consider are:

- Measure module current consumption in a specified operating state. Deviations from expected results (compared to a "Golden Device") can signal potential issues.

- Perform functional tests, including communication checks with the host controller and verification of interfaces.
- Assess fundamental RF characteristics (modulation accuracy, power levels, spectrum). Verify that the device meets expected performance standards.

23. Physical specifications

23.1. Dimensions

| Dimensions |
|--------------|
| 9 x 7 x 2 mm |

Table 29: Dimensions

23.2. Weight

| Weight |
|--------|
| < 1 g |

Table 30: Weight

23.3. Light sensitivity

Inside the Proteus-e a light sensitive WLCSP package is used. This package is sensitive to visible and near infrared light. As the chip is not completely shielded on the sides, any mounting without enclosure could lead to malfunction. This should be taken into account when designing an enclosure for the end device.

23.4. Module drawing

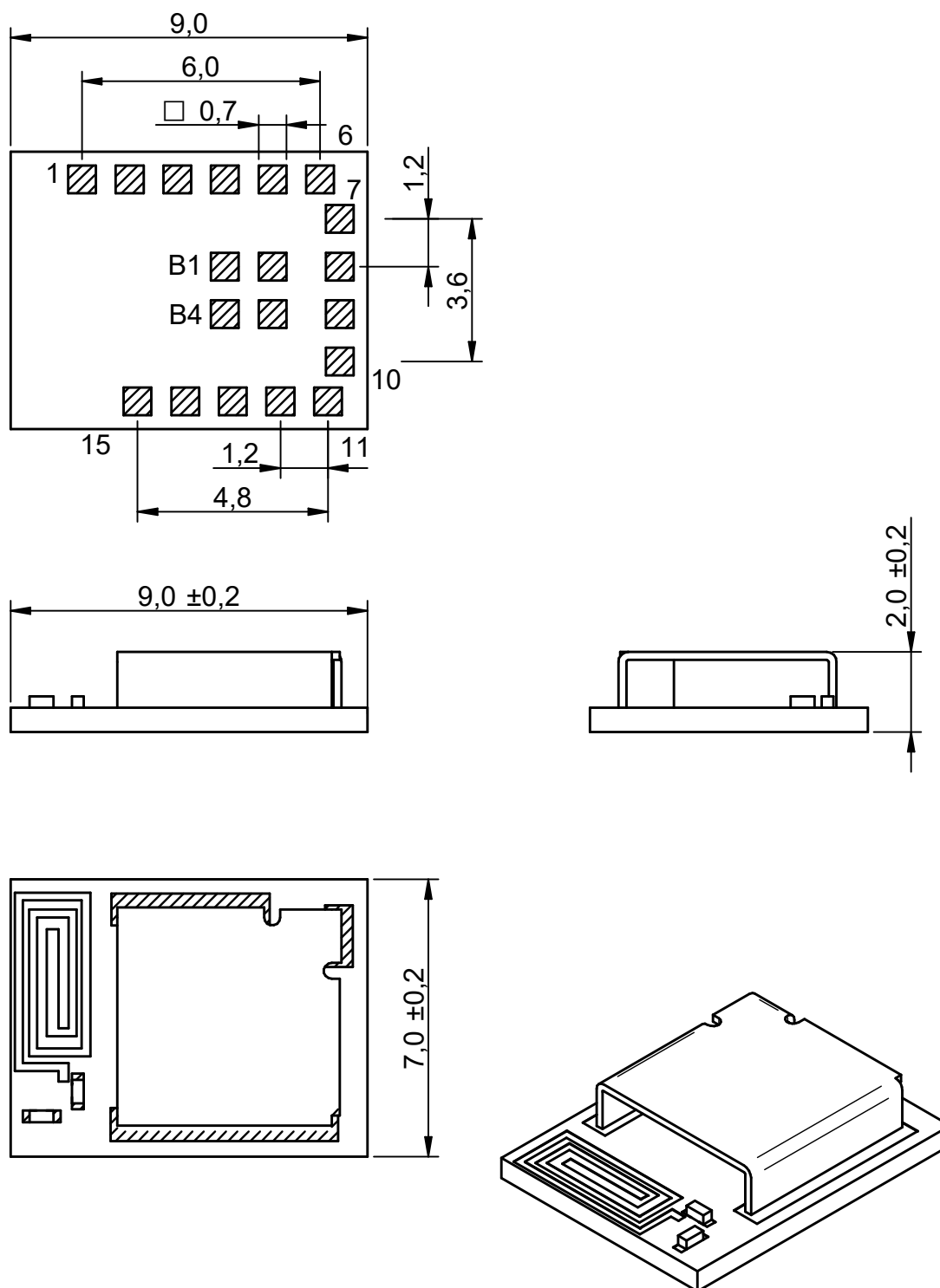


Figure 40: Module dimensions [mm]

23.5. Footprint WE-FP-4+

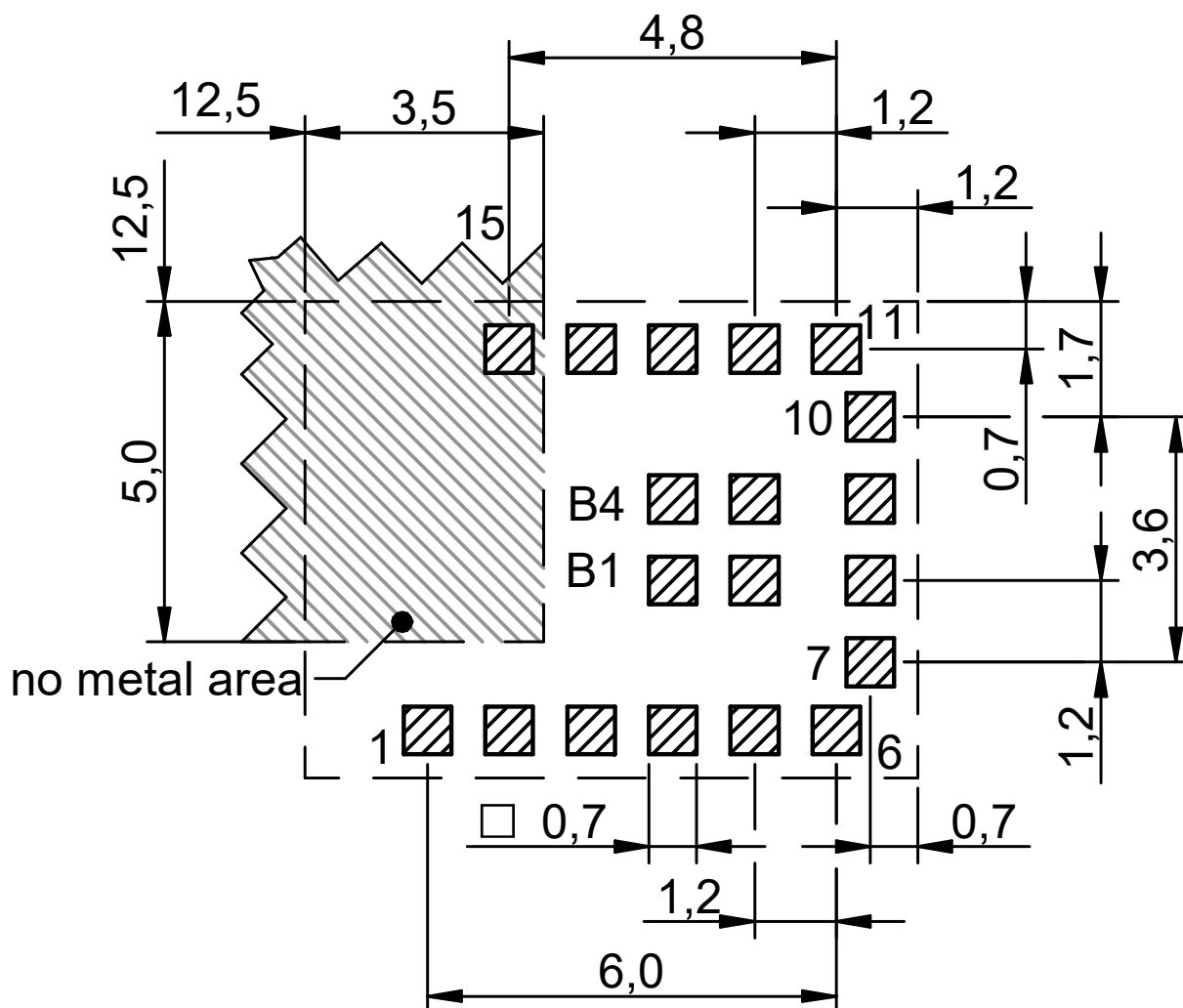


Figure 41: Footprint WE-FP-4+ [mm]

23.6. Antenna free area

To avoid influence and mismatching of the antenna the recommended free area around the antenna should be maintained. As rule of thumb a minimum distance of metal parts to the antenna of $\lambda/10$ should be kept (see figure 41). Even though metal parts would influence the characteristic of the antenna, but the direct influence and matching keep an acceptable level.

24. Marking

24.1. Lot number

The 15 digit lot number is printed in numerical digits as well as in form of a machine readable bar code. It is divided into 5 blocks as shown in the following picture and can be translated according to the following table.

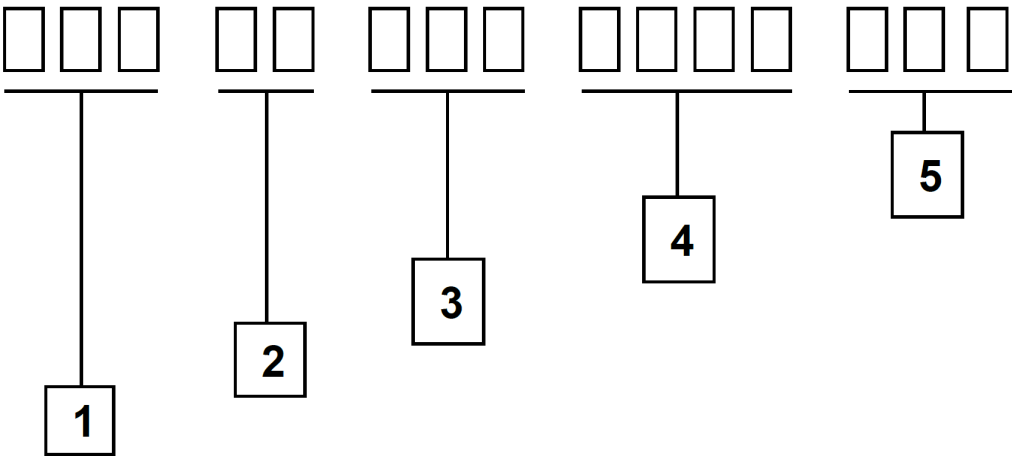


Figure 42: Lot number structure

| Block | Information | Example(s) |
|-------|---|---|
| 1 | eiSos internal, 3 digits | 438 |
| 2 | eiSos internal, 2 digits | 01 |
| 3 | Radio module hardware version, 3 digits | V2.4 = 024, V12.2 = 122 |
| 4 | Date code, 4 digits | 1703 = week 03 in year 2017, 1816 = week 16 in year 2018 |
| 5 | Radio module firmware version, 3 digits | V3.2 = 302, V5.13 = 513 |

Table 31: Lot number details

As the user can perform a firmware update the printed lot number only shows the factory delivery state. The currently installed firmware can be requested from the module using the corresponding product specific command. The firmware version as well as the hardware version are restricted to show only major and minor version not the patch identifier. Block 5 is not applicable for products without firmware.

24.2. General labeling information

Labels of Würth Elektronik eiSos radio modules include several fields. Besides the manufacturer identification, the product's *WE* order code, serial number and certification information are placed on the label. In case of small labels, additional certification marks are placed on the label of the reel.

The information on the label are fixed. Only the serial number changes with each entity of the radio module. For Proteus-e the label is as follows:

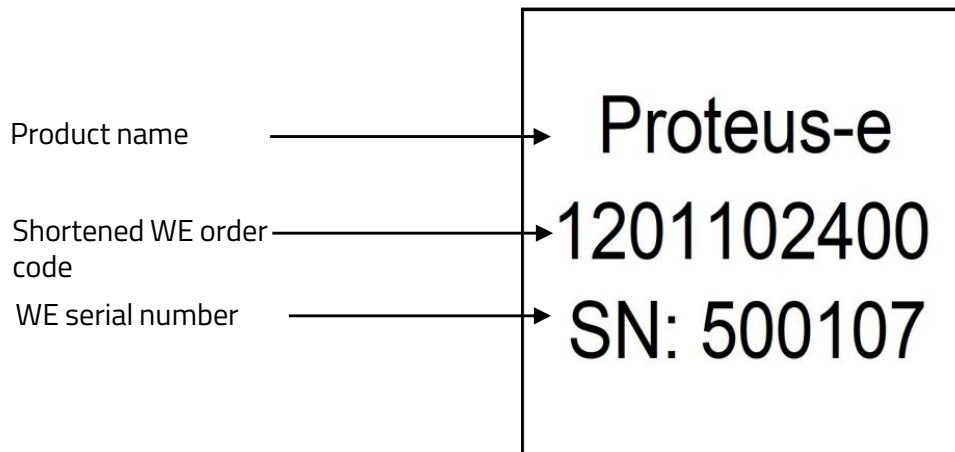


Figure 43: Label of the Proteus-e

25. Information for explosion protection

In case the end product should be used in explosion protection areas the following information can be used:

- The module itself is unfused.
- The maximum output power of the module is 5 dBm for radio pad
- The total amount of capacitance of all capacitors is 6.8 μF .
- The total amount of inductance of all inductors is 10.009 μH .
- A DC/DC regulator is included in the chip set and used to obtain low power functionality.

26. References

- [1] Bluetooth®. Bluetooth® Core Specification, version 5.1. <https://www.bluetooth.com/specifications/specs/core-specification-5-1/>.
- [2] Würth Elektronik. Application note 24 - Proteus-e advanced developer guide. <http://www.we-online.com/ANR024>.
- [3] Würth Elektronik. Application note 25 - Proteus-e quick start. <http://www.we-online.com/ANR025>.
- [4] Eddystone Beacon. [https://en.wikipedia.org/wiki/Eddystone_\(Google\)](https://en.wikipedia.org/wiki/Eddystone_(Google)).
- [5] iBeacon. <https://en.wikipedia.org/wiki/IBeacon>.
- [6] Würth Elektronik. Application note 26 - Proteus beacon. <http://www.we-online.com/ANR026>.
- [7] Würth Elektronik. WE UART Terminal PC tool (Smart Commander). <https://www.we-online.de/wcs-software>.
- [8] Würth Elektronik. Wireless Connectivity SDK for Raspberry Pi - Radio module drivers in C-code. <https://github.com/WurthElektronik/WirelessConnectivity-SDK>.
- [9] Würth Elektronik. Wireless Connectivity SDK for STM32 - Radio module drivers in C-code. https://github.com/WurthElektronik/WirelessConnectivity-SDK_STM32.
- [10] IEEE Registration Authority. <https://standards.ieee.org/products-programs/regauth/>.
- [11] Würth Elektronik. Himalia. https://www.we-online.com/catalog/en/WIRL_ACCE_2600130021.
- [12] Würth Elektronik. Application note 27 - Bluetooth listing guide. <http://www.we-online.com/ANR027>.

27. Bluetooth SIG listing/qualification

| Type | Data |
|-----------------------|-------------|
| Design name | Proteus-e |
| Declaration ID | D057370 |
| QDID | 177585 |
| Specification version | 5.1 |
| Project type | End product |

Each product containing intellectual property of the Bluetooth® Special Interest Group (SIG) must be qualified by the SIG to obtain the corresponding Declaration ID.

Due to the qualification of the Proteus-e as end product no further Bluetooth® tests are required. The only arising expenses are those for purchasing a Bluetooth® Declaration ID.

To obtain the Bluetooth® listing of the end device, please refer to the application note ANR027 [12].

28. Regulatory compliance information

28.1. Important notice EU

The use of RF frequencies is limited by national regulations. The Proteus-e has been designed to comply with the RED directive 2014/53/EU of the European Union (EU).

The Proteus-e can be operated without notification and free of charge in the area of the European Union. However, according to the RED directive, restrictions (e.g. in terms of duty cycle or maximum allowed RF power) may apply.

Modifications (2014/53/EU article 3 (i))

Caution: Changes or modifications for this equipment not expressly approved by Würth Elektronik eiSos may void the CE conformity to operate this equipment.



Since the module itself is not fused the voltage supply shall be fed from a power source which is class PS2 according to EN 62368-1.

28.2. Important notice UKCA

The UK's government has laid legislation to continue recognition of current EU requirements for a range of product regulations, including the CE marking. The Radio Equipment Regulation 2017/1206 is within the scope of this announcement, among others.

Consequently, the Proteus-e can be sold and utilized in the UK with the CE marking, without the need of UKCA declaration of conformity or UKCA marking.

Source: <https://www.gov.uk/guidance/ce-marking>

28.3. Important notice FCC

The use of RF frequencies is limited by national regulations. The Proteus-e has been designed to comply with the FCC Part 15.

The Proteus-e can be operated without notification and free of charge in the area of the United States of America. However, according to the FCC Part 15, restrictions (e.g. in terms of maximum allowed RF power and antenna) may apply.

28.4. Conformity assessment of the final product

The Proteus-e is a subassembly. It is designed to be embedded into other products (products incorporating the Proteus-e are henceforward referred to as "final products").

It is the responsibility of the manufacturer of the final product to ensure that the final product is in compliance with the essential requirements of the underlying national radio regulations.

The conformity assessment of the subassembly Proteus-e carried out by Würth Elektronik eiSos does not replace the required conformity assessment of the final product.

28.5. Exemption clause

Relevant regulation requirements are subject to change. Würth Elektronik eiSos does not guarantee the accuracy of the before mentioned information. Directives, technical standards, procedural descriptions and the like may be interpreted differently by the national authorities. Equally, the national laws and restrictions may vary with the country. In case of doubt or uncertainty, we recommend that you consult with the authorities or official certification organizations of the relevant countries. Würth Elektronik eiSos is exempt from any responsibilities or liabilities related to regulatory compliance.

Notwithstanding the above, Würth Elektronik eiSos makes no representations and warranties of any kind related to their accuracy, correctness, completeness and/or usability for customer applications. No responsibility is assumed for inaccuracies or incompleteness.

28.6. EU Declaration of conformity



EU DECLARATION OF CONFORMITY

Radio equipment: 2612011024000

The manufacturer: Würth Elektronik eiSos GmbH & Co. KG
Max-Eyth-Straße 1
74638 Waldenburg

This declaration of conformity is issued under the sole responsibility of the manufacturer.

Object of the declaration: 2612011024000

The object of the declaration described above is in conformity with the relevant Union harmonisation legislation Directive 2014/53/EU. Following harmonised norms or technical specifications have been applied:

EN 300 328 V2.2.2 (2019-07)
EN 301 489-1 V2.2.3 (2019-11)
EN 301 489-17 V3.3.1 (2024-09)
EN 62479 : 2010
EN 62368-1:2014 + AC:2015
2011/65/EU with its amending Annex II EU 2015/863

i.A. G. Eschardt

Trier, 12th of January 2026

Place and date of issue

28.7. RED-DA Cybersecurity statement

Cybersecurity as per articles 3.3d, 3.3e and 3.3f of the Radio Equipment Directive Delegated Act. The RED-DA mandates to comply to the EN 18031-1, 18031-2 and 18031-3 in order to fulfill the requirements of the cybersecurity chapters (d, e and f).

- EN 18031-1: Common security requirements for radio equipment - Part 1: Internet connected radio equipment
- EN 18031-2: Common security requirements for radio equipment - Part 2: Radio equipment processing data, namely internet connected radio equipment, childcare radio equipment, toys radio equipment and wearable radio equipment
- EN 18031-3: Common security requirements for radio equipment - Part 3: Internet connected radio equipment processing virtual money or monetary value

| Requirements | Statement and conditions |
|--|---|
| (d) Radio equipment does not harm the network or its functioning nor misuses network resources, thereby causing an unacceptable degradation of service | <p>"Not applicable": The product is not capable itself to communicate over the internet. The product is only able to communicate via the following protocols and interfaces. None of the protocols contained in the product are "internet-connectable".</p> <p>Radio communication protocols: This Bluetooth (2.1, 4.x, 5.x, 6.x) product does not support or include the "Internet Protocol Support Profile". Bluetooth is a set of radio standards (e.g. Bluetooth Classic, Bluetooth Low Energy, Bluetooth MESH, Bluetooth LE Audio, ...) maintained by the Bluetooth SIG.</p> <p>Host Interface, wired: The host interface of the product does not support internet connectivity. UART is used as a wired communication and control channel towards the customers host.</p> |
| (e) Radio equipment incorporates safeguards to ensure that the personal data and privacy of the user and of the subscriber are protected | <p>"Not applicable": The product is not internet connected. The product does not pose a risk to the users or subscribers privacy, as it does not store or process any personal data.</p> |
| (f) Radio equipment supports certain features ensuring protection from fraud | <p>"Not applicable": The product is not internet connected. The product does not pose a risk of fraud because it does not store or process financial data or enables financial transactions.</p> |

28.8. FCC Compliance Statement (US)

FCC ID: R7T1201102

This device complies with Part 15 of the FCC Rules.

Operation is subject to the following two conditions:

- (1) this device may not cause harmful interference, and
 - (2) this device must accept any interference received, including interference that may cause undesired operation.
- (FCC 15.19)

Modifications (FCC 15.21)

Caution: Changes or modifications for this equipment not expressly approved by Würth Elektronik eiSos may void the FCC authorization to operate this equipment.

28.8.1. FCC certificate

| <p style="text-align: center; font-weight: bold; color: blue;">TCB</p> | <p>GRANT OF EQUIPMENT AUTHORIZATION</p> <p>Certification Issued Under the Authority of the Federal Communications Commission</p> <p>By:</p> <p>CTC advanced GmbH Untertürkheimer Strasse 6-10 66117 Saarbruecken, Germany</p> <p>Date of Grant: 11/26/2021 Application Dated: 11/26/2021</p> | <p style="text-align: center; font-weight: bold; color: blue;">TCB</p> | | | | | | | | | | |
|---|---|--|-----------------------|---------------------|---------------------|---------------------|-----|-----------------|---------|--|--|--|
| <p>Wuerth Elektronik eiSos GmbH & Co KG Max-Eyth-Strasse 1 Waldenburger, 74638 Germany</p> <p>Attention: Gudrun Eckhardt, Manager</p> | | | | | | | | | | | | |
| <p>NOT TRANSFERABLE</p> <p>EQUIPMENT AUTHORIZATION is hereby issued to the named GRANTEE, and is VALID ONLY for the equipment identified hereon for use under the Commission's Rules and Regulations listed below.</p> | | | | | | | | | | | | |
| <p>FCC IDENTIFIER: R7T1201102</p> <p>Name of Grantee: Wuerth Elektronik eiSos GmbH & Co KG</p> <p>Equipment Class: Digital Transmission System</p> <p>Notes: Bluetooth LE radio module</p> <p>Modular Type: Single Modular</p> | | | | | | | | | | | | |
| <p><u>Grant Notes</u></p> <p>Output power listed is peak conducted.</p> <p>Module integrators have to follow the grantee's module integration guide.</p> | <table border="1" style="width: 100%; border-collapse: collapse; font-size: small;"> <thead> <tr> <th style="text-align: left;">FCC Rule Parts</th> <th style="text-align: left;">Frequency Range (MHz)</th> <th style="text-align: left;">Output Watts</th> <th style="text-align: left;">Frequency Tolerance</th> <th style="text-align: left;">Emission Designator</th> </tr> </thead> <tbody> <tr> <td>15C</td> <td>2402.0 ~ 2480.0</td> <td>0.08302</td> <td></td> <td></td> </tr> </tbody> </table> | FCC Rule Parts | Frequency Range (MHz) | Output Watts | Frequency Tolerance | Emission Designator | 15C | 2402.0 ~ 2480.0 | 0.08302 | | | |
| FCC Rule Parts | Frequency Range (MHz) | Output Watts | Frequency Tolerance | Emission Designator | | | | | | | | |
| 15C | 2402.0 ~ 2480.0 | 0.08302 | | | | | | | | | | |

Figure 44: FCC certificate

28.9. IC Compliance Statement (Canada)

Certification Number: 5136A-1201102

HVIN: 1201102

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes : (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

28.9.1. IC certificate

Technical Acceptance Certificate - Canada

 CTC advanced GmbH
 Untertuerkheimer Str. 6-10 | 66117 Saarbruecken
 Germany | www.ctcadvanced.com



advanced
member of RWTÜV group

Certificate Holder: Würth Elektronik eiSos GmbH & Co
 Max-Eyth-Str. 1
 74638 Waldenburg
 Germany

ISED Certification Number: 5136A-1201102

CTC Registration Number: 1-1754/21-04

Wireless Test Lab Number: 3462C

Wireless Test Lab: CTC advanced GmbH
 Untertuerkheimer Strasse 6-10
 66117 Saarbruecken
 Germany
 Phone: +49 681 598 0
 Fax: +49 681 598 8775
 E-Mail: info@ctcadvanced.com
 Web: www.ctcadvanced.com

Product Description: Bluetooth LE radio module

Certification of equipment means only that the equipment has met the requirements of the above-noted specification. Licence applications, where applicable to use certified equipment, are acted on accordingly by the ISED issuing office and will depend on the existing radio environment, service and location of operation. This certificate is issued on condition that the holder complies and will continue to comply with the requirements and procedures issued by ISED. The equipment for which this certificate is issued shall not be manufactured, imported, distributed, leased, offered for sale or sold unless the equipment complies with the applicable technical specifications and procedures issued by ISED.

La certification du matériel signifie seulement que le matériel a satisfait aux exigences de la norme indiquée ci-dessus. Les demandes de licences nécessaires pour l'utilisation du matériel certifié sont traitées en conséquence par le bureau de délivrance d'ISDE et dépendent des conditions radio ambiantes, du service et de l'emplacement d'exploitation. Le présent certificat est délivré à la condition que le titulaire satisfasse et continue de satisfaire aux exigences et aux procédures d'ISDE. Le matériel à l'égard duquel le présent certificat est délivré ne doit pas être fabriqué, importé, distribué, loué, mis en vente ou vendu à moins d'être conforme aux procédures et aux spécifications techniques applicables publiées par ISDE.

I hereby attest that the subject equipment was tested and found in compliance with the above noted specification.
 J'atteste, par la présente, que le matériel a fait l'objet d'essai et a été jugé conforme à la spécification ci-dessus.



Deutsche
Akkreditierungsstelle
D-35 12076-61 000

Based on the Comprehensive Economic
and Trade Agreement (CETA) between
the European Community and Canada.



FOREIGN CERTIFICATION BODY
CAB ID NO DE0001

CTC advanced GmbH
 Untertuerkheimer Str. 6-10 | 66117 Saarbruecken
 Germany

CTC advanced GmbH
 c/o Gerald Schmidt, c/o CTC
 advanced GmbH
 our-SCT-211118
 email=gerald.schmidt@ctcadvanced.com, c/o DE
 2021.11.26.12:03:32+01'00'

Saarbruecken
 Certification Body

Figure 45: IC certificate

28.10. FCC and IC requirements to OEM integrators

This module has been granted modular approval. OEM integrators for host products may use the module in their final products without additional FCC/IC (Industry Canada) certification if they meet the following conditions. Otherwise, additional FCC/IC approvals must be obtained. The host product with the module installed must be evaluated for simultaneous transmission requirements.

- The users manual for the host product must clearly indicate the operating requirements and conditions that must be observed to ensure compliance with current FCC/IC RF exposure guidelines.
- A label must be affixed to the outside of the host product with the following statements:
This device contains FCC ID: R7T1201102
This equipment contains equipment certified under IC ID: 5136A-1201102
- The final host / module combination may also need to be evaluated against the FCC Part 15B criteria for unintentional radiators in order to be properly authorized for operation as a Part 15 digital device.
- If the final host / module combination is intended for use as a portable device (see classifications below) the host manufacturer is responsible for separate approvals for the SAR requirements from FCC Part 2.1093 and RSS-102.

28.10.1. OEM requirements:

The OEM must ensure that the following conditions are met.

- The Proteus-e will be used at a distance of at least 10 mm to the human body.
- End users of products, which contain the module, must not have the ability to alter the firmware that governs the operation of the module. The agency grant is valid only when the module is incorporated into a final product by OEM integrators.
- The end-user must not be provided with instructions to remove, adjust or install the module.
- The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product. Attaching a label to a removable portion of the final product, such as a battery cover, is not permitted.
- The label must include the following text:
Contains FCC ID: R7T1201102
The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:
(i.) this device may not cause harmful interference and
(ii.) this device must accept any interference received, including interference that may cause undesired operation.

When the device is so small or for such use that it is not practicable to place the statement

above on it, the information required by this paragraph shall be placed in a prominent location in the instruction manual or pamphlet supplied to the user or, alternatively, shall be placed on the container in which the device is marketed. However, the FCC identifier or the unique identifier, as appropriate, must be displayed on the device.

- The user manual for the end product must also contain the text given above.
 - Changes or modifications not expressly approved could void the user's authority to operate the equipment.
 - The module must be used with only the following approved antenna(s).

28.10.2. Pre-certified antennas

The Proteus-e is pre-certified with the following antennas.

| Product | Certified antenna |
|---------------------------|---------------------------------------|
| Proteus-e (2612011024000) | PCB antenna included in the Proteus-e |

28.11. TELEC radio law approval (Japan)

Japanese Radio Law Compliance



This device has passed the Radio Law approval for Japan through the registered certification body TELEC. The corresponding ARIB (Association of Radio Industries and Businesses) standard has been applied. Accordingly, the market approval is given by the MIC (Ministry of Internal Affairs and Communications).

ID-Code
(Interference
provision)

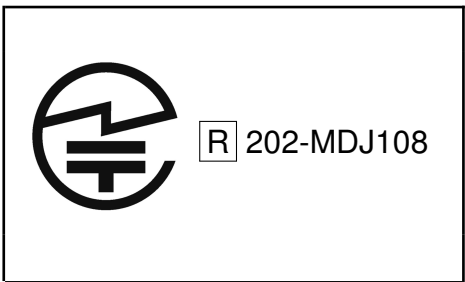
This device should not be modified (otherwise the granted designation number will become invalid)

The MAC address of the radio device maintains the format 00:18:DA:xx:xx:xx. The latter part xx:xx:xx of the MAC address coincides with the serial number of the device.

28.11.1. Label

Due to the size of the Proteus-e label, the certification label of the Proteus-e is not placed onto the module label.

2612011024000:



After integration of the Proteus-e in the end device, the corresponding certification label must be recognizable from the outside. Otherwise this information must be referenced on the housing as well as in the user manual. E labeling is allowed.

28.11.2. Certified antennas

The Proteus-e is pre-certified with the following antennas.

| Product | Certified antenna |
|---------------------------|--|
| Proteus-e (2612011024000) | PCB antenna included in the Proteus-e |
| Proteus-e (2612011024000) | Dipole antenna with maximum gain of 2.8 dBi (2600130021) |

28.11.3. TELEC certificate

| | | |
|---|---|--|
| CTC advanced GmbH Untertürkheimer Strasse 6-10, D-66117 Saarbrücken, Germany | | CTC  advanced <small>member of RWTUV group</small> |
| Conformity Assessment Body Registered Certification Body for Japan | | |
| 認証書 TYPE - BASED CERTIFICATE | | |
| Approval Holder 認証を受けた者 | Würth Elektronik eiSos GmbH & Co KG Max-Eyth-Str. 1 74638, Waldenburg Germany | |
| Manufacturer Name 製造者名 | Würth Elektronik eiSos GmbH & Co KG Max-Eyth-Str. 1 74638, Waldenburg Germany | |
| Model Name 型式又は名称 | 2612011024000 | |
| Classification of Specified Radio equipment 特定無線設備の種類 | Ordinance concerning Technical Regulations Conformity Certification etc. of Specified Radio Equipment 特定無線設備の技術基準適合証明等に関する規則 | |
| Type of Emissions, Frequency and Antenna Power 電波の形式、周波数及び空中線電力 | 第2条第1項第19号に掲げる無線設備 Low power data communications system in the 2.4GHz band F1D 2402 ~ 2480MHz (2MHz separation, 40 channels) 0.003W | |
| Certified Number 認証番号 | 202-MDJ108 | |
| Certified Date 認証年月日 | 2021-12-16 | |
| 上記のとおり、電波法第 38 条の 24 第 1 項の規定に基づく認証を行ったものであることを証する。 This is to certify that the above-mentioned certification by type has been granted in accordance with the provisions of Article 38-24, Paragraph 1 of the Radio Law. | | |
| Recognized by The Ministry of Internal Affairs and Communications (MIC) via Bundesnetzagentur BNetzA-CAB-03/25-21  | | CTC advanced GmbH Stefan Mann <i>Signature</i>  |
| | | cn=Stefan Mann, o=CTC advanced GmbH, ou=MAN-201022, email=stefan.mann@ctc-advanced.com, c=DE 2021.12.16 13:38:20 +01'00' |

Figure 46: TELEC certificate


¹Additional, not yet certified, antennas must be re-certified without retesting. Only antenna gain and antenna characteristic diagrams must be specified. Please contact your Business Development Engineer (BDM) to get support in certifying your own antenna.


28.12. ETA-WPC (India)

Registration No: ETA-SD-20230807136 Date: 25-08-2023

The Proteus-e complies with the provisions on the Equipment Type Approval WPC Wing for India.

28.12.1. ETA-WPC certificate





सत्यमेव जयते

Government of India
Ministry of Communications
Department of Telecommunications
WPC Wing
Sanchar Bhawan, New Delhi-110001.

[Generation of Equipment Type Approval (ETA) through self-declaration issued under O.M. No. ETA-WPC /Policy/2018-19 dated 26 February, 2019].

THIS ETA IS ISSUED FOR A SINGLE MODEL WITH MODEL NAME Proteus-e (2612011024000)

Registration No: ETA-SD-20230807136 Date: 25-08-2023

I). Details of Applicant and Parameters of Equipment:

| | | |
|----|--|--|
| 1. | Name & Address of the first Applicant. (Indian Manufacturer/ Authorised Indian representative for foreign manufacturer) | WURTH ELECTRONICS SERVICES INDIA PRIVATE LIMITED, Ground and 1st Floor, No. 3, Prestige Sterling Square, Madras Bank Road, Next to Airlines Hotel, Bangalore, Bengaluru Bangalore Urban, Karnataka, 560001, Bangalore Urban,KARNATAKA,560001 |
| 2. | Equipment category | Bluetooth Low Energy Module |
| 3. | Make | Wurth Elektronik eiSos GmbH & Co. KG,Germany |
| 4. | Model | Proteus-e (2612011024000) |
| 5. | Frequency range(s) of Equipment | 1. 2402-2480 MHz |
| 6. | Max output power/Field strength/PSD | 1. E.I.R.P. (dBm). -4 |

1 / 2

Figure 47: ETA-WPC certificate page 1

| | | | |
|----|--|--|------------------------------|
| 7. | Applicable Gazette Notification(s) | 1. 45 (E) Dated 28-01-2005 | |
| 8. | RF Test Report details:- | | |
| | Name&Address /Country of accredited laboratory issuing the RF test report | Accreditation Certificate Reference/Number | Test Report No. and Date |
| | CTC advanced GmbH & Untertuerkheimer Strasse 6 10 66117 Saarbruecken / Germany | D-PL-12076-01-03 | 1-1754/21-04-02 & 21-09-2021 |

II). Terms and Conditions

- (i). This certificate will not be valid in case any change in the above parameters and not conforming to the Gazette Notification mentioned in sl.no 7 above.
- (ii). Use of such equipment has been exempted from licensing requirement vide Gazette Notification mentioned in sl. no. 7, on Non-interference,Non-protectionand sharing (non-exclusive) basis.
- (iii). Use of such equipment in case not conforming to above notification will require a specific wireless operating license, as applicable from this Ministry.
- (iv). Field units of WPC Wing reserve the right for sample check/audit carried out for the purpose of RF analysis/spectrum monitoring in view to avoid interference to other wireless users and ensure compliance of technical parameters mentioned in sl no. 5,6&7.
- (v). This certificate is valid only for equipment which are exempted from import licensing requirements as per the Import Policy of DGFT and for import of such device, a self-declaration based, system generated (Saralsanchar) Import undertaking/ permission is required.
- (vi). The applicant is liable for prosecution under Indian Law in case of any wrong declaration/ submission of ingenuine RF test report(s) for issue of ETA through Self-Declaration.

Note:

1. Once ETA through self-declaration is generated for a model, subsequently it may be utilized by other person(s) for import/usage purpose in India.
2. The importers of above model shall comply with other import related requirements, if any, with Customs.

This is Self-generated certificate. Hence, no signature is required. It may be downloaded/verified from the website <https://saralsanchar.gov.in>.

Figure 48: ETA-WPC certificate page 2


28.13. NCC Compliance Statement (Taiwan)

NCC Certification ID: CCAF23Y10130T2

The Proteus-e complies with the provisions on the National Communications Commission of Taiwan.

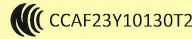
Due to the small size of the Proteus-e label the NCC Certification ID is placed on the packaging label.

28.13.1. NCC certificate



耕興股份有限公司
電信管制射頻器材型式認證證明

證照字號：型式字第 AF 號

| | |
|---------------|---|
| 一、申請者： | 伍仕電子股份有限公司 |
| 二、地址： | 臺北市內湖區民權東路6段18之1號4樓 |
| 三、製造廠商： | WE, Würth Elektronik or Würth Elektronik eiSos |
| 四、器材名稱： | Bluetooth 5.1 module T&R |
| 五、廠牌： | Würth Electronics Co., Ltd. |
| 六、型號： | 2612011024000 |
| 七、發射功率（電場強度）： | 詳如備註表格 |
| 八、工作頻率： | 詳如備註表格 |
| 九、審驗日期： | 112年8月3日 |
| 十、審驗合格標籤式樣： |  |

十一、警語或標示要求：（器材本體、使用手冊、外包裝盒等應遵守下列標示要求）

- 請依上列型號、標籤式樣於電信管制射頻器材本體明顯處標示其型號及審驗合格標籤，並於包裝盒標示主管機關印章。最終產品應於本體明顯處標示非隨插即用射頻模組（組件）之審驗合格標籤及最終產品型號，並於包裝盒標示主管機關印章。
- 電信管制射頻器材應依本會或相關技術規範規定於指定位置標示中文警語。
- 經授權使用射頻模組（組件）之審驗合格標籤者，應於最終產品說明書及包裝盒提供充分與正確之資訊。
- 於國際網路販賣取得審驗證明之電信管制射頻器材者，應於該國際網路網頁提供審驗合格標籤或符合性聲明標籤資訊。
- 使用手冊應標示下列資訊：
 - 取得審驗證明之低功率射頻器材，非經核准，公司、商號或使用者均不得擅自變更頻率、加大功率或變更原設計之特性及功能。低功率射頻器材之使用不得影響飛航安全及干擾合法通信；經發現有干擾現象時，應立即停用，並改善至無干擾時方得繼續使用。前述合法通信，指依電信管理法規定作業之無線電通信。低功率射頻器材須忍受合法通信或工業、科學及醫療用電波輻射性電機設備之干擾。

十二、特殊記載事項：

- 經取得審驗證明之電信管制射頻器材，如變更其廠牌、型號、技術規格或射頻性能時，除電信管制射頻器材審驗管理辦法另有規定外，應重新申請審驗。
- 經取得審驗證明之審驗證明之電信管制射頻器材或非隨插即用射頻模組（組件），於電信管制射頻器材相關技術規範修訂審驗相關章節時，應依技術規範明定之實施期限，申請重新審驗辦理，未依規定重新審驗者，原審驗機關（構）得廢止其審驗證明。
- 取得型式認證證明、符合性聲明證明或簡易符合性聲明證明者，應妥善保管申請審驗之電信管制射頻器材或非隨插即用射頻模組（組件）、外接電源、配件、外接天線、與檢驗報告或測試報告相符之測試治具及與檢驗報告或測試報告使用相同版本之測試軟體至該器材停止生產或停止輸入後五年。
- 取得型式認證證明或符合性聲明證明者得授權他人於同廠牌同型號之電信管制射頻器材或非隨插即用射頻模組（組件）使用審驗合格標籤或符合性聲明標籤，授權他人使用審驗合格標籤或符合性聲明標籤以取得審驗證明者於主管機關指定網站登錄或委託原審驗機關（構）登錄。
- 以取得審驗證明之完全射頻模組（組件）組裝成完全最終產品後，取得該完全射頻模組（組件）之審驗證明者，應於該完全最終產品販賣前，檢附標註完全最終產品廠牌、型號及外觀照片之電子檔案並應提供組裝該完全射頻模組（組件）之完全最終產品其內部及電路板4x6吋以上具尺規之至少二面彩色照片，電路板主要元件須清晰可辨識。向原審驗機關（構）登錄。

說明：

- （一）本公司係依電信管理法第87條第2項及第3項規定訂定之電信管制射頻器材測試機構及驗證機構管理辦法第4條第1項規定，經認證組織（財團法人全國認證基金會）認可產品驗證制度符合 CNS 17065 或 ISO/IEC 17065 標準（TAF 標準編號：PC006），並經國家通訊傳播委員會委託辦理電信管制射頻器材之審驗工作。（機構地址：桃園市龜山區華亞一路52號、電話：03-3273456），核發本型式認證證明。
- （二）本設備之製造或輸入須遵守電信管理法相關規定。

(續下頁，續頁分開使用無效)

第 1 頁 共 2 頁

Figure 49: NCC certificate page 1

(三) 配件資訊：

| 名稱 | 廠牌/型號 | 備註 |
|-----|-------|----|
| 充電器 | 無此配件 | - |
| 鋰電池 | 無此配件 | - |
| 充電線 | 無此配件 | - |
| 其 他 | 無此配件 | - |

備註：

1. 依「商品標示法」及「電器及電子商品標示基準」規定，標示事項貼於商品或內外包裝上，以免違法而受處分。

2. 本型式認證證明僅代表完成電信管制射頻器材審驗管理辦法所訂之審驗程序其審驗範圍僅限本器材之無線射頻硬體功能，不作為其他(如器材之資通安全檢測、產地..等)認可證明。

3. 本器材電信介面使用天線資訊如下：

| Ant. | 廠牌 | 型號 | 天線型式 | 天線增益 |
|---------|------|---------------|-------------|----------|
| Antenna | 伍仕電子 | 2612011024000 | PCB antenna | -7.9 dBi |

4. 本設備之電信介面符合技術規範及工作頻段資訊如下。

| 電信介面 | 技術規範(年份) | 章節 | 工作頻率 | 功率 or 場強 | 備註 |
|-----------|----------------|------|--------------|----------|---------------|
| Bluetooth | LP0002(109 年版) | 4.10 | 2402~2480MHz | 4.30 dBm | BT LE GFSK |

BW=Bandwidth

中 華 民 國 1 1 2 年 8 月 3 日

以下空白

第 2 頁 共 2 頁

Figure 50: NCC certificate page 2

28.14. Certification of the end device

For the certification of the end device, which integrates the Proteus-e, it is necessary to run the Bluetooth® direct test modes (DTM) on the Proteus-e and check its radio behavior.

To do that, the Proteus-e standard firmware provides already the required direct test mode commands (see chapter 9.5). These commands follow the radio module's command interface, and must be sent via UART to the Proteus-e.



For running the certification tests, the radio module's UART pins must be exposed to externally control the DTM functions.

In case the test house requires to use a Bluetooth® tester², the radio module must support the 2-wire DTM protocol on the UART. In this scenario it is necessary to flash the radio module with a 2-wire DTM firmware using the production flash interface mentioned in chapter 16.1.



In case it is required to flash the 2-wire DTM firmware, the SWD pins be exposed as well to use the production programming interface.

The 2-wire DTM firmware is provided on request, as well as the DTM GUI PC tool. This tool is used to send the DTM test commands to the radio module. It supports both protocols, the standard firmware DTM as well as the 2-wire DTM.



Figure 51: Bluetooth® DTM GUI PC tool

²A Bluetooth® tester is a certified device running automated tests on a connected Bluetooth® enabled device.

29. Important notes

The following conditions apply to all goods within the wireless connectivity and sensors product range of Würth Elektronik eiSos GmbH & Co. KG:

General customer responsibility

Some goods within the product range of Würth Elektronik eiSos GmbH & Co. KG contain statements regarding general suitability for certain application areas. These statements about suitability are based on our knowledge and experience of typical requirements concerning the areas, serve as general guidance and cannot be estimated as binding statements about the suitability for a customer application. The responsibility for the applicability and use 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, where appropriate to investigate and to decide whether the device with the specific product characteristics described in the product specification is valid and suitable for the respective customer application or not. Accordingly, the customer is cautioned to verify that the documentation is current before placing orders.

Customer responsibility related to specific, in particular safety-relevant applications

It has to be clearly pointed out that the possibility of a malfunction of electronic components or failure before the end of the usual lifetime cannot be completely eliminated in the current state of the art, even if the products are operated within the range of the specifications. The same statement is valid for all software source code and firmware parts contained in or used with or for products in the wireless connectivity and sensor product range of Würth Elektronik eiSos GmbH & Co. KG. In certain customer applications requiring a high level of safety and especially in customer applications in which the malfunction or failure of an electronic component could endanger human life or health, it must be ensured by most advanced technological aid of suitable design of the customer application that no injury or damage is caused to third parties in the event of malfunction or failure of an electronic component.

Best care and attention

Any product-specific data sheets, manuals, application notes, PCNs, warnings and cautions must be strictly observed in the most recent versions and matching to the products revisions. These documents can be downloaded from the product specific sections on the wireless connectivity and sensors homepage.

Customer support for product specifications

Some products within the product range may contain substances, which are subject to restrictions in certain jurisdictions in order to serve specific technical requirements. Necessary information is available on request. In this case, the Business Development Engineer (BDM) or the internal sales person in charge should be contacted who will be happy to support in this matter.

Product improvements

Due to constant product improvement, product specifications may change from time to time. As a standard reporting procedure of the Product Change Notification (PCN) according to the JEDEC-Standard, we inform about major changes. In case of further queries regarding the PCN, the Business Development Engineer (BDM), the internal sales person or the technical support team in charge should be contacted. The basic responsibility of the customer as per section 29 and 29 remains unaffected.

All software like "wireless connectivity SDK", "Sensor SDK" or other source codes as well as all PC software tools are not subject to the Product Change Notification information process.

Product life cycle

Due to technical progress and economical evaluation, we also reserve the right to discontinue production and delivery of products. As a standard reporting procedure of the Product Termination Notification (PTN) according to the JEDEC-Standard we will inform at an early stage about inevitable product discontinuance. According to this, we cannot ensure that all products within our product range will always be available. Therefore, it needs to be verified with the Business Development Engineer (BDM) or the internal sales person in charge about the current product availability expectancy before or when the product for application design-in disposal is considered. The approach named above does not apply in the case of individual agreements deviating from the foregoing for customer-specific products.

Property rights

All the rights for contractual products produced by Würth Elektronik eiSos GmbH & Co. KG on the basis of ideas, development contracts as well as models or templates that are subject to copyright, patent or commercial protection supplied to the customer will remain with Würth Elektronik eiSos GmbH & Co. KG. Würth Elektronik eiSos GmbH & Co. KG does not warrant or represent that any license, either expressed or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, application, or process in which Würth Elektronik eiSos GmbH & Co. KG components or services are used.

General terms and conditions

Unless otherwise agreed in individual contracts, all orders are subject to the current version of the "General Terms and Conditions of Würth Elektronik eiSos Group", last version available at www.we-online.com.

30. Legal notice

Exclusion of liability

Würth Elektronik eiSos GmbH & Co. KG considers the information in this document to be correct at the time of publication. However, Würth Elektronik eiSos GmbH & Co. KG reserves the right to modify the information such as technical specifications or functions of its products or discontinue the production of these products or the support of one of these products without any written announcement or notification to customers. The customer must make sure that the information used corresponds to the latest published information. Würth Elektronik eiSos GmbH & Co. KG does not assume any liability for the use of its products. Würth Elektronik eiSos GmbH & Co. KG does not grant licenses for its patent rights or for any other of its intellectual property rights or third-party rights.

Notwithstanding anything above, Würth Elektronik eiSos GmbH & Co. KG makes no representations and/or warranties of any kind for the

provided information related to their accuracy, correctness, completeness, usage of the products and/or usability for customer applications. Information published by Würth Elektronik eiSos GmbH & Co. KG regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof.

Suitability in customer applications

The customer bears the responsibility for compliance of systems or units, in which Würth Elektronik eiSos GmbH & Co. KG products are integrated, with applicable legal regulations. Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of Würth Elektronik eiSos GmbH & Co. KG components in its applications, notwithstanding any applications-related information or support that may be provided by Würth Elektronik eiSos GmbH & Co. KG. Customer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences lessen the likelihood of failures that might cause harm and take appropriate remedial actions. The customer will fully indemnify Würth Elektronik eiSos GmbH & Co. KG and its representatives against any damages arising out of the use of any Würth Elektronik eiSos GmbH & Co. KG components in safety-critical applications.

Trademarks

AMBER wireless is a registered trademark of Würth Elektronik eiSos GmbH & Co. KG. All other trademarks, registered trademarks, and product names are the exclusive property of the respective owners.

Usage restriction

Würth Elektronik eiSos GmbH & Co. KG products have been designed and developed for usage in general electronic equipment only. This product is not authorized for use in equipment where a higher safety standard and reliability standard is especially required or where a failure of the product is reasonably expected to cause severe personal injury or death, unless the parties have executed an agreement specifically governing such use. Moreover, Würth Elektronik eiSos GmbH & Co. KG products are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation (automotive control, train control, ship control), transportation signal, disaster prevention, medical, public information network etc. Würth Elektronik eiSos GmbH & Co. KG 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 electronic component, which is used in electrical circuits that require high safety and reliability function or performance. By using Würth Elektronik eiSos GmbH & Co. KG products, the customer agrees to these terms and conditions.

31. License terms

These License terms will take effect upon the purchase and usage of the Würth Elektronik eiSos GmbH & Co. KG wireless connectivity products. You hereby agree that these license terms are applicable to the product and the incorporated software, firmware and source codes (collectively, "Software") made available by Würth Elektronik eiSos in any form, including but not limited to binary, executable or source code form. The software included in any Würth Elektronik eiSos wireless connectivity product is purchased to you on the condition that you accept the terms and conditions of these license terms. You agree to comply with all provisions under these license terms.

Limited license

Würth Elektronik eiSos hereby grants you a limited, non-exclusive, non-transferable and royalty-free license to use the software and under the conditions that will be set forth in these license terms. You are free to use the provided software only in connection with one of the products from Würth Elektronik eiSos to the extent described in these license terms. You are entitled to change or alter the source code for the sole purpose of creating an application embedding the Würth Elektronik eiSos wireless connectivity product. The transfer of the source code to third parties is allowed to the sole extent that the source code is used by such third parties in connection with our product or another hardware provided by Würth Elektronik eiSos under strict adherence of these license terms. Würth Elektronik eiSos will not assume any liability for the usage of the incorporated software and the source code. You are not entitled to transfer the source code in any form to third parties without prior written consent of Würth Elektronik eiSos.

You are not allowed to reproduce, translate, reverse engineer, decompile, disassemble or create derivative works of the incorporated software and the source code in whole or in part. No more extensive rights to use and exploit the products are granted to you.

Usage and obligations

The responsibility for the applicability and use of the Würth Elektronik eiSos wireless connectivity product with the incorporated firmware in a particular customer design is always solely within the authority of the customer. Due to this fact, it is up to you to evaluate and investigate, where appropriate, and to decide whether the device with the specific product characteristics described in the product specification is valid and suitable for your respective application or not.

You are responsible for using the Würth Elektronik eiSos wireless connectivity product with the incorporated firmware in compliance with all applicable product liability and product safety laws. You acknowledge to minimize the risk of loss and harm to individuals and bear the risk for failure leading to personal injury or death due to your usage of the product.

Würth Elektronik eiSos' products with the incorporated firmware are not authorized for use in safety-critical applications, or where a failure of the product is reasonably expected to cause severe personal injury or death. Moreover, Würth Elektronik eiSos' products with the incorporated firmware are neither designed nor intended for use in areas such as military, aerospace, aviation, nuclear control, submarine, transportation (automotive control, train control, ship control), transportation signal, disaster prevention, medical, public information network etc. You shall inform Würth Elektronik eiSos about the intent of such usage before design-in stage. In certain customer applications requiring a very high level of safety and in which the malfunction or failure of an electronic component could endanger human life or health, you must ensure to have all necessary expertise in the safety and regulatory ramifications of your applications. You acknowledge and agree that you are solely responsible for all legal, regulatory and safety-related requirements concerning your products and any use of Würth Elektronik eiSos' products with the incorporated firmware in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by Würth Elektronik eiSos. **YOU SHALL INDEMNIFY WÜRTH ELEKTRONIK EISOS AGAINST ANY DAMAGES ARISING OUT OF THE USE OF WÜRTH ELEKTRONIK EISOS' PRODUCTS WITH THE INCORPORATED FIRMWARE IN SUCH SAFETY-CRITICAL APPLICATIONS.**

Ownership

The incorporated firmware created by Würth Elektronik eiSos is and will remain the exclusive property of Würth Elektronik eiSos.

Firmware update(s)

You have the opportunity to request the current and actual firmware for a bought wireless connectivity product within the time of warranty. However, Würth Elektronik eiSos has no obligation to update a modules firmware in their production facilities, but can offer this as a service on request. The upload of firmware updates falls within your responsibility, e.g. via ACC or another software for firmware updates. Firmware updates will not be communicated automatically. It is within your responsibility to check the current version of a firmware in the latest version of the product manual on our website. The revision table in the product manual provides all necessary information about firmware updates. There is no right to be provided with binary files, so called "firmware images", those could be flashed through JTAG, SWD, Spi-Bi-Wire, SPI or similar interfaces.

Disclaimer of warranty

THE FIRMWARE IS PROVIDED "AS IS". YOU ACKNOWLEDGE THAT WÜRTH ELEKTRONIK 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 YOUR INTENDED PURPOSE OR USAGE. WÜRTH ELEKTRONIK 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 THE WÜRTH ELEKTRONIK EISOS' PRODUCT WITH THE INCORPORATED FIRMWARE IS USED. INFORMATION PUBLISHED BY WÜRTH ELEKTRONIK EISOS REGARDING THIRD-PARTY PRODUCTS OR SERVICES DOES NOT CONSTITUTE A LICENSE FROM WÜRTH ELEKTRONIK EISOS TO USE SUCH PRODUCTS OR SERVICES OR A WARRANTY OR ENDORSEMENT THEREOF.

Limitation of liability

Any liability not expressly provided by Würth Elektronik eiSos shall be disclaimed.

You agree to hold us harmless from any third-party claims related to your usage of the Würth Elektronik eiSos' products with the incorporated firmware, software and source code. Würth Elektronik eiSos disclaims any liability for any alteration, development created by you or your customers as well as for any combination with other products.

Applicable law and jurisdiction

Applicable law to these license terms shall be the laws of the Federal Republic of Germany. Any dispute, claim or controversy arising out of or relating to these license terms shall be resolved and finally settled by the court competent for the location of Würth Elektronik eiSos registered office.

Severability clause

If a provision of these license terms is or becomes invalid, unenforceable or null and void, this shall not affect the remaining provisions of the terms. The parties shall replace any such provisions with new valid provisions that most closely approximate the purpose of the terms.

Miscellaneous

Würth Elektronik eiSos reserves the right at any time to change these terms at its own discretion. It is your responsibility to check at Würth Elektronik eiSos homepage for any updates. Your continued usage of the products will be deemed as the acceptance of the change.

We recommend you to be updated about the status of new firmware and software, which is available on our website or in our data sheet and manual, and to implement new software in your device where appropriate.

By ordering a product, you accept these license terms in all terms.

List of Figures

| | | |
|-----|--|-----|
| 1. | Compatible central devices implementing the Bluetooth® LE profiles of Proteus-e | 14 |
| 2. | Proteus-e | 14 |
| 3. | Block diagram of the module | 17 |
| 4. | Sleep current (no RAM retention, wake on reset) over operating temperature range | 20 |
| 5. | Radio transmitting @ 4 dBm output power, 1 Mbps Bluetooth® LE mode, Clock = HFXO, Regulator = DC/DC (typical values) | 21 |
| 6. | Current consumption calculation in advertising mode with 40 ms advertising interval with 4 dBm output power and 31 Bytes radio data, UART disabled | 22 |
| 7. | Pinout (top view) | 25 |
| 8. | Minimal pin connections | 27 |
| 9. | Power up | 30 |
| 10. | State overview | 34 |
| 11. | Power up | 36 |
| 12. | Steps for the connection setup | 38 |
| 13. | DTM flow chart | 41 |
| 14. | Command sequence when transmitting data | 137 |
| 15. | Switch of the <i>BUSY</i> pin when transmitting data | 140 |
| 16. | Handling the <i>/RTS</i> and <i>BUSY</i> pin | 141 |
| 17. | Configure the local GPIOs via local host | 143 |
| 18. | Configure the local GPIOs via remote device host | 143 |
| 19. | Read the configuration of the local GPIOs via local host | 144 |
| 20. | Read the configuration of the local GPIOs via remote device host | 144 |
| 21. | Set the output value of a GPIO via host controller | 145 |
| 22. | Read the input value of a GPIO via host controller | 145 |
| 23. | Set the output value of a GPIO via remote device | 146 |
| 24. | Read the input value of a GPIO via remote device | 146 |
| 25. | Layout | 155 |
| 26. | Placement of the module with integrated antenna | 156 |
| 27. | Dimensioning the antenna connection as micro strip | 156 |
| 28. | Himalia dipole antenna | 159 |
| 29. | Reference design: Schematic | 161 |
| 30. | Top layer (top), bottom layer (bottom) | 162 |
| 31. | Antenna characteristic from integrated antenna measured on official EV-Board ¹ | 163 |
| 32. | Stack-up | 164 |
| 33. | Simple short schematic | 165 |
| 34. | Simple short layout | 166 |
| 35. | Capacitor internal antenna schematic | 166 |
| 36. | Capacitor internal antenna layout | 167 |
| 37. | Capacitor external antenna schematic | 168 |
| 38. | Capacitor external antenna layout | 168 |
| 39. | Reflow soldering profile | 171 |
| 40. | Module dimensions [mm] | 177 |
| 41. | Footprint WE-FP-4+ [mm] | 178 |
| 42. | Lot number structure | 179 |
| 43. | Label of the Proteus-e | 180 |
| 44. | FCC certificate | 188 |

| | | |
|-----|--------------------------------------|-----|
| 45. | IC certificate | 189 |
| 46. | TELEC certificate | 193 |
| 47. | ETA-WPC certificate page 1 | 194 |
| 48. | ETA-WPC certificate page 2 | 195 |
| 49. | NCC certificate page 1 | 196 |
| 50. | NCC certificate page 2 | 197 |
| 51. | Bluetooth® DTM GUI PC tool | 198 |

List of Tables

| | | |
|-----|--|-----|
| 3. | Ordering information | 17 |
| 4. | Operating conditions | 18 |
| 5. | Absolute maximum ratings | 18 |
| 6. | Current consumption - transmitting | 19 |
| 7. | Current consumption - receiving | 19 |
| 8. | Current consumption - low power | 19 |
| 9. | RSSI accuracy | 23 |
| 10. | Timing | 23 |
| 11. | Transmit and receive power | 23 |
| 12. | Sensitivity at different data rates | 23 |
| 13. | Pin characteristics | 24 |
| 14. | Pinout | 26 |
| 15. | LED behavior of the Proteus-e | 35 |
| 16. | Message overview: Requests | 93 |
| 17. | Message overview: Confirmations | 94 |
| 18. | Message overview: Indications | 94 |
| 19. | nRF52805 IC revision overview | 96 |
| 20. | Security configuration flags | 106 |
| 22. | Table of user settings | 135 |
| 23. | Table of runtime settings | 135 |
| 24. | Maximum throughput timings, packet error rate = 0% | 138 |
| 25. | Supported GPIO_IDs | 147 |
| 26. | UUID default values | 148 |
| 28. | Classification reflow soldering profile, Note: refer to IPC/JEDEC J-STD-020E | 170 |
| 29. | Dimensions | 176 |
| 30. | Weight | 176 |
| 31. | Lot number details | 179 |
| 33. | CRC8 Test Vectors | 204 |

A. Additional CRC8 Information

This Annex gives an example CRC8 implementation and test vectors.

A.1. Example CRC8 Implementation

```
#include <stdint.h>

uint8_t Get_CRC8(uint8_t * bufP, uint16_t len)
{
    uint8_t crc = 0x00;
    for (uint16_t i = 0; i < len; i++)
    {
        crc ^= bufP[i];
    }
    return crc;
}
```

Code 1: Example CRC8 Implementation

A.2. CRC8 Test Vectors

| Input data | Data length | Resulting CRC8 |
|--|-------------|----------------|
| Null | 0 | 0x00 |
| 0x02 0x01 0x00 0x00 | 4 | 0x03 |
| 0x02 0x87 0x01 0x00 0x16 | 5 | 0x92 |
| 0x02 0x04 0x04 0x00 0x41 0x42 0x43 0x44 | 8 | 0x06 |
| 0x02 0x88 0x07 0x00 0x00 0x55 0x00 0x00 0xDA 0x18 0x00 | 11 | 0x1A |

Table 33: CRC8 Test Vectors

B. Example code for host integration

The following code is an example implementation of a function to transmit data using a 2 byte length field in the command frame. For demonstration reasons, the Proteus-III has been taken. The full function codes of all radio modules are available in the Wireless Connectivity SDK (www.we-online.com/wco-SDK).

```
#define CMD_PAYLOAD_MAX 964
typedef struct {
    uint8_t Stx;
    uint8_t Cmd;
    uint16_t Length;           /* LSB first */
    uint8_t Data[CMD_PAYLOAD_MAX+1]; /* +1 for CRC8 */
} CMD_Frame_t;

#define CMD_OFFSET_TO_DATAFIELD 4
#define CMD_OVERHEAD (CMD_OFFSET_TO_DATAFIELD+1)

bool ProteusIII_Transmit(uint8_t *PayloadP, uint16_t length)
{
    /* fill request message with STX, command byte and length field */
    CMD_Frame_t CMD_Frame;
    CMD_Frame.Stx = CMD_STX; /* 0x02 */
    CMD_Frame.Cmd = ProteusIII_CMD_DATA_REQ; /* 0x04 */
    CMD_Frame.Length = length;

    /* fill request message with user payload */
    memcpy(CMD_Frame.Data, PayloadP, length);

    /* fill request message with CRC8 */
    CMD_Frame.Data[CMD_Frame.Length] = Get_CRC8(&CMD_Frame, CMD_Frame.Length +
        CMD_OFFSET_TO_DATAFIELD);

    /* transmit full message via UART to radio module */
    UART_SendBytes(&CMD_Frame, (CMD_Frame.Length + CMD_OVERHEAD));

    /* wait for response message from radio module */
    return UART_Wait_for_Response(CMD_WAIT_TIME, ProteusIII_CMD_TXCOMPLETE_RSP,
        CMD_Status_Success, true);
}
```

Code 2: Example function implementation for radio modules with 2 byte length field

**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