



USER MANUAL

THYONE-E

2612011021000

VERSION 1.0

OCTOBER 17, 2024

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.

Revision history

| Manual version | FW version | HW version | Notes | Date |
|----------------|------------|------------|---|-------------|
| 1.0 | 1.9 | 1.0 | <ul style="list-style-type: none">Initial release | August 2024 |

* For firmware history see chapter [Firmware history](#)

Abbreviations

| Abbreviation | Name | Description |
|---------------|---|--|
| BDM | Business Development Manager | Support and sales contact person responsible for limited sales area. |
| CS | Checksum | Byte wise XOR combination of the preceding fields. |
| ETX | End-of-Text | Characters that signal end of a sequence. |
| GFSK | Gaussian Frequency Shift Keying | Modulation scheme. |
| I/O | Input/Output | Pinout description. |
| LPM | Low Power Mode | Mode for efficient power consumption. |
| Payload | | The intended message in a frame / package. |
| PRBS | Pseudorandom Binary Sequence | Binary sequence generated by a deterministic random generator. |
| 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. |
| 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 9. |
| UART | Universal Asynchronous Receiver Transmitter | Allows the serial communication with the module. |
| [HEX] 0xhh | Hexadecimal | All numbers beginning with 0x are hexadecimal numbers. All other numbers are decimal, unless stated otherwise. |

Contents

| | |
|--|-----------|
| Overview of helpful application notes | 9 |
| 1. Introduction | 10 |
| 1.1. Key features | 10 |
| 1.2. Block diagram | 12 |
| 1.3. Ordering information | 12 |
| 2. Electrical specifications | 13 |
| 2.1. Recommended operating conditions | 13 |
| 2.2. Absolute maximum ratings | 13 |
| 2.3. Power consumption | 14 |
| 2.4. Radio characteristics | 16 |
| 2.5. Pin characteristics | 17 |
| 3. Pinout | 18 |
| 4. Quick start | 20 |
| 4.1. Minimal pin connections | 20 |
| 4.2. Antenna connection | 22 |
| 4.2.1. On-board PCB antenna | 22 |
| 4.2.2. External antenna | 22 |
| 4.3. Power up | 23 |
| 4.4. Quick start example | 23 |
| 4.4.1. Prerequisites | 23 |
| 4.4.2. Hardware configuration | 23 |
| 4.4.3. Set-up description | 23 |
| 4.4.4. Start-up | 23 |
| 4.4.5. Transmit and receive data | 25 |
| 5. Functional description | 28 |
| 5.1. Radio | 28 |
| 5.1.1. Radio profiles | 28 |
| 5.1.2. RF channel | 29 |
| 5.1.3. RF transmit power | 29 |
| 5.1.4. Radio compatibility to Thyone-I | 29 |
| 5.2. Low power mode | 29 |
| 5.3. Network addressing | 29 |
| 5.4. Acknowledgment and retries | 30 |
| 5.5. Repeater mode | 30 |
| 5.6. Packet sniffer mode | 30 |
| 5.7. Encryption/decryption | 30 |
| 5.8. Local/remote digital I/O control | 31 |
| 5.9. Transparent mode | 31 |
| 5.10. Radio test mode | 31 |
| 5.11. Modes of operation | 32 |

| | |
|--|-----------|
| 6. Host connection | 34 |
| 6.1. Serial interface: UART | 34 |
| 6.1.1. Timing and reset behaviour | 35 |
| 7. The command interface | 36 |
| 7.1. Command categories | 36 |
| 7.2. Command structure | 36 |
| 7.3. User data commands | 37 |
| 7.3.1. CMD_BROADCAST_DATA_REQ | 37 |
| 7.3.2. CMD_MULTICAST_DATA_REQ | 38 |
| 7.3.3. CMD_UNICAST_DATA_REQ | 38 |
| 7.3.4. CMD_MULTICAST_DATA_EX_REQ | 38 |
| 7.3.5. CMD_UNICAST_DATA_EX_REQ | 39 |
| 7.3.6. CMD_DATA_CNF | 39 |
| 7.3.7. CMD_TXCOMPLETE_RSP | 39 |
| 7.3.8. CMD_DATA_IND | 39 |
| 7.3.9. CMD_SNIFFER_IND | 40 |
| 7.4. Configuring the module | 41 |
| 7.4.1. CMD_SET_REQ | 41 |
| 7.4.1.1. Example 1 | 42 |
| 7.4.2. CMD_GET_REQ | 42 |
| 7.4.2.1. Example 1 | 42 |
| 7.5. Manage the device state | 44 |
| 7.5.1. CMD_START_IND | 44 |
| 7.5.2. CMD_GETSTATE_REQ | 44 |
| 7.5.3. CMD_RESET_REQ | 45 |
| 7.5.4. CMD_SLEEP_REQ | 45 |
| 7.5.5. CMD_FACTORY_RESET_REQ | 46 |
| 7.5.6. CMD_TRANSPARENT_MODE_REQ | 46 |
| 7.5.7. CMD_SETCHANNEL_REQ | 47 |
| 7.6. Digital I/O control | 48 |
| 7.6.1. CMD_GPIO_LOCAL_SET_CONFIG_REQ | 48 |
| 7.6.1.1. Example: Configure two GPIOs to output high | 49 |
| 7.6.2. CMD_GPIO_LOCAL_GET_CONFIG_REQ | 50 |
| 7.6.2.1. Example: Read the current GPIO configuration | 50 |
| 7.6.3. CMD_GPIO_REMOTE_SET_CONFIG_REQ | 52 |
| 7.6.3.1. Example: Configure two GPIOs of the remote device to output HIGH | 53 |
| 7.6.4. CMD_GPIO_REMOTE_GET_CONFIG_REQ | 55 |
| 7.6.4.1. Example: Read the current GPIO configuration of the remote device | 56 |
| 7.6.5. CMD_GPIO_LOCAL_WRITE_REQ | 57 |
| 7.6.5.1. Example: Set a local output GPIO to LOW | 58 |
| 7.6.6. CMD_GPIO_LOCAL_READ_REQ | 59 |
| 7.6.6.1. Example: Read the values of local GPIOs | 59 |
| 7.6.7. CMD_GPIO_REMOTE_WRITE_REQ | 61 |
| 7.6.7.1. Example: Set a remote output GPIO to LOW | 62 |
| 7.6.8. CMD_GPIO_REMOTE_READ_REQ | 63 |
| 7.6.8.1. Example: Read the values of remote GPIOs | 64 |

| | | |
|-----------|---|-----------|
| 7.6.9. | CMD_GPIO_REMOTE_SET_CONFIG_IND | 65 |
| 7.6.9.1. | Example: Two GPIOs have been configured by the remote device 0x6C000001 to output HIGH | 65 |
| 7.6.10. | CMD_GPIO_REMOTE_WRITE_IND | 66 |
| 7.6.10.1. | Example: GPIOs have been written via remote access | 66 |
| 7.7. | Other messages | 67 |
| 7.7.1. | CMD_ERROR_IND | 67 |
| 7.8. | Run the radio test modes | 68 |
| 7.8.1. | CMD_DTM_START_REQ | 68 |
| 7.8.2. | CMD_DTM_REQ | 68 |
| 7.8.2.1. | Example: Transmission, 16 times 0x0F, channel 0 | 70 |
| 7.8.2.2. | Example: Receiver, channel 0 | 71 |
| 7.8.2.3. | Example: Transmission, carrier test, channel 0 | 71 |
| 7.8.2.4. | Example: Set TX power to -4 dBm | 72 |
| 7.8.2.5. | Example: Set PHY to 2 Mbit/s mode | 72 |
| 7.9. | Messages Overview | 73 |
| 8. | The transparent interface | 76 |
| 8.1. | Entering the transparent mode | 76 |
| 8.2. | Leaving the transparent mode | 76 |
| 8.2.1. | Transparent mode escape sequence | 76 |
| 8.3. | Transparent mode configuration | 77 |
| 8.4. | Restrictions in transparent mode | 77 |
| 9. | User settings - Module configuration values | 79 |
| 9.1. | SERIAL_NUMBER: Read the serial number of the module | 79 |
| 9.1.1. | Example 1 | 79 |
| 9.2. | FW_Version: Read the firmware version | 80 |
| 9.2.1. | Example 1 | 80 |
| 9.3. | UART_CONFIG: Modify the UART speed | 81 |
| 9.3.1. | Example 1 | 83 |
| 9.3.2. | Example 2 | 83 |
| 9.4. | UART_MODE: Mode of UART operation | 84 |
| 9.4.1. | Example 1 | 84 |
| 9.4.2. | Example 2 | 84 |
| 9.5. | UART_TRANSPARENT_TIMEOUT: Mode of UART operation | 85 |
| 9.5.1. | Example 1 | 85 |
| 9.5.2. | Example 2 | 85 |
| 9.6. | RF_CHANNEL: Channel (Frequency) of operation | 86 |
| 9.6.1. | Example 1 | 86 |
| 9.6.2. | Example 2 | 87 |
| 9.7. | ENCRYPTION_MODE: Set encryption mode | 88 |
| 9.7.1. | Example 1 | 88 |
| 9.7.2. | Example 2 | 88 |
| 9.8. | RF_PROFILE: Modify the radio profile | 90 |
| 9.8.1. | Example 1 | 90 |
| 9.8.2. | Example 2 | 90 |

| | | |
|---------|--|-----|
| 9.9. | RF_NUM_RETRIES: Number of retries | 91 |
| 9.9.1. | Example 1 | 91 |
| 9.9.2. | Example 2 | 91 |
| 9.10. | RF_TX_POWER: Modify the output power | 92 |
| 9.10.1. | Example 1 | 92 |
| 9.10.2. | Example 2 | 92 |
| 9.11. | RF_REPEATER_THRESHOLD: Modify the repeater threshold | 93 |
| 9.11.1. | Example 1 | 93 |
| 9.11.2. | Example 2 | 93 |
| 9.12. | RF_RP_NUM_SLOTS: Number of retries | 94 |
| 9.12.1. | Example 1 | 95 |
| 9.12.2. | Example 2 | 95 |
| 9.13. | MAC_SOURCE_ADDRESS: Modify the source address of the module | 96 |
| 9.13.1. | Example 1 | 96 |
| 9.13.2. | Example 2 | 96 |
| 9.14. | MAC_DEST_ADDRESS: Modify the default destination address | 97 |
| 9.14.1. | Example 1 | 97 |
| 9.14.2. | Example 2 | 97 |
| 9.15. | MAC_GROUP_ID: Modify the default group ID | 98 |
| 9.15.1. | Example 1 | 98 |
| 9.15.2. | Example 2 | 98 |
| 9.16. | MAC_TRANSPARENT_ADDR_MODE: Modify the addressing mode used in transparent mode | 99 |
| 9.16.1. | Example 1 | 99 |
| 9.16.2. | Example 2 | 99 |
| 9.17. | MAC_ENCRYPTION_KEY: Set the key used for encryption/decryption | 100 |
| 9.17.1. | Example 1 | 100 |
| 9.18. | MAC_TTL: Time to live | 101 |
| 9.18.1. | Example 1 | 101 |
| 9.18.2. | Example 2 | 101 |
| 9.19. | GPIO_BLOCK_REMOTE_GPIO_CONFIG | 102 |
| 9.19.1. | Example 1 | 102 |
| 9.19.2. | Example 2 | 102 |
| 9.20. | UART_TRANSP_ETX_CONFIG: Trigger radio transmit on ETX characters | 103 |
| 9.20.1. | Example 1 | 103 |
| 9.20.2. | Example 2 | 103 |
| 9.21. | UART_TRANSP_ETX: ETX characters | 105 |
| 9.21.1. | Example 1 | 105 |
| 9.21.2. | Example 2 | 105 |
| 9.22. | UART_TRANSP_ESC_ENABLE: Enable switch to command mode on escape sequence | 106 |
| 9.22.1. | Example 1 | 106 |
| 9.22.2. | Example 2 | 106 |
| 9.23. | UART_TRANSP_ESC: Escape sequence | 107 |
| 9.23.1. | Example 1 | 107 |
| 9.23.2. | Example 2 | 107 |
| 9.24. | MODULE_MODE: Mode of radio operation | 108 |
| 9.24.1. | Example 1 | 108 |

| | |
|---|------------|
| 9.24.2. Example 2 | 108 |
| 10.Remote GPIO control | 111 |
| 10.1. Supported GPIOs for remote and local control | 117 |
| 11.Flooding mesh: Using the repeater functionality | 118 |
| 11.1. Setup of the network and repeater device | 119 |
| 11.2. Example network | 120 |
| 11.2.1. Application in parallel networks | 121 |
| 12.Timing parameters | 122 |
| 12.1. Start-up | 122 |
| 12.2. Data transmission and throughput measurements | 124 |
| 12.2.1. Command mode | 124 |
| 12.2.2. Transparent mode | 125 |
| 13.Custom firmware | 127 |
| 13.1. Custom configuration of standard firmware | 127 |
| 13.2. Customer specific firmware | 127 |
| 13.3. Customer firmware | 127 |
| 13.4. Contact for firmware requests | 128 |
| 14.Firmware history | 129 |
| 14.1. Release notes | 129 |
| 15.Design in guide | 130 |
| 15.1. Advice for schematic and layout | 130 |
| 15.2. Designing the antenna connection | 132 |
| 15.3. Antenna solutions | 133 |
| 15.3.1. Wire antenna | 134 |
| 15.3.2. Chip antenna | 134 |
| 15.3.3. PCB antenna | 134 |
| 15.3.4. Antennas provided by Würth Elektronik eiSos | 135 |
| 15.3.4.1. 2600130021 - Himalia dipole antenna | 135 |
| 16.Reference design | 136 |
| 16.1. EV-Board | 137 |
| 16.1.1. Schematic | 137 |
| 16.2. Layout | 138 |
| 16.3. Internal antenna radiation characteristics | 139 |
| 16.4. Trace design | 140 |
| 16.4.1. Simple short using internal antenna | 141 |
| 16.4.2. 22 pF coupling capacitor using internal antenna | 142 |
| 16.4.3. 22 pF coupling capacitor using external antenna | 144 |
| 16.5. Antenna fine tuning | 145 |
| 17.Manufacturing information | 146 |
| 17.1. Moisture sensitivity level | 146 |

| | |
|--|------------|
| 17.2. Soldering | 146 |
| 17.2.1. Reflow soldering | 146 |
| 17.2.2. Cleaning | 147 |
| 17.2.3. Potting and coating | 148 |
| 17.2.4. Other notations | 148 |
| 17.3. ESD handling | 148 |
| 17.4. Safety recommendations | 149 |
| 18. Physical specifications | 150 |
| 18.1. Dimensions | 150 |
| 18.2. Weight | 150 |
| 18.3. Module drawing | 151 |
| 18.4. Footprint WE-FP-4+ | 152 |
| 18.5. Antenna free area | 152 |
| 19. Marking | 153 |
| 19.1. Lot number | 153 |
| 19.2. General labeling information | 154 |
| 20. Information for explosion protection | 155 |
| 21. References | 156 |
| 22. Regulatory compliance information | 157 |
| 22.1. Important notice EU | 157 |
| 22.2. Important notice FCC | 157 |
| 22.3. Conformity assessment of the final product | 157 |
| 22.4. Exemption clause | 157 |
| 22.5. EU Declaration of conformity | 158 |
| 22.6. FCC Compliance Statement (US) | 159 |
| 22.7. IC Compliance Statement (Canada) | 159 |
| 22.8. FCC and IC requirements to OEM integrators | 159 |
| 22.8.1. OEM requirements: | 160 |
| 22.8.2. Pre-certified antennas | 160 |
| 22.9. ETA-WPC (India) | 162 |
| 22.9.1. ETA-WPC certificate | 162 |
| 23. Important notes | 164 |
| 24. Legal notice | 164 |
| 25. License terms | 165 |
| A. Additional CRC8 Information | 169 |
| A.1. Example CRC8 Implementation | 169 |
| A.2. CRC8 Test Vectors | 169 |
| B. Example code for host integration | 170 |

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 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.

Application note ANR031 - Certification of custom modules

<http://www.we-online.com/ANR031>

This application note explains how certifications of a standard product can be used to gain the certification of a customized product. This is done for firmware, that has been adapted by Würth Elektronik eiSos, as well as for firmware written by customer.

1. Introduction

The Thyone-e module is a radio sub-module for wireless communication between devices such as control systems, remote controls, sensor nodes etc. Operating in the globally available 2.4 GHz license free band, Thyone-e offers a robust and secure data transmission in point-to-point as well as mesh configurations.

The Thyone-e is flashed with the WE-ProWare radio stack, which ensures high flexibility without compromising the reliability. Interfacing with the host system via serial UART, the module allows easy configuration and control of the radio using a simple command interface. To ensure ease-of-use for cable replacement applications, the module also offers a transparent mode to function as a serial-to-radio adapter. Small dimensions comparable to a nano-SIM card, including an on-board PCB antenna, makes Thyone-e ideal for small form factor design.

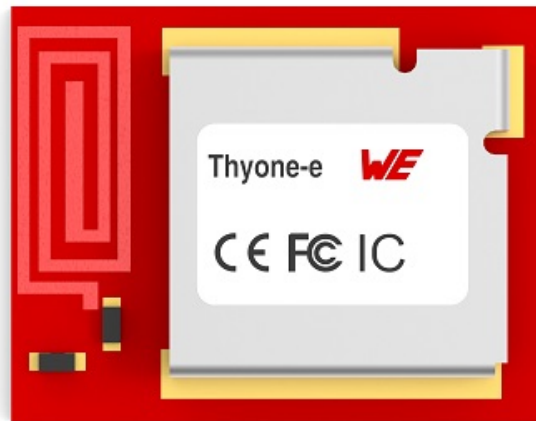


Figure 1: Thyone-e

1.1. Key features

The Thyone-e offers a wide range of configurable features to suit even the most sophisticated application designs.

Extremely small dimensions: Due to its small size (9 x 7 mm), the module can be easily designed-in to compact end devices.

Energy efficient: The Thyone-e has extremely low current consumption, especially in sleep mode (< 0.4 μ A), making it suitable for battery-driven applications. More information can be found in section 2.3.

Globally available 2.4 GHz band: The Thyone-e operates in the 2.4 GHz license free band, allowing a global deployment of the end-device.

Smart antenna selection: The Thyone-e offers a choice of using the on-board PCB antenna for compact designs or connecting an external antenna for applications that require longer range.

High throughput mode: The Thyone-e offers a radio profile with 2 Mbit/s data transmission over the air, leading to an effective end-to-end throughput of around 275 kbit/s.

Fast serial interface: The Thyone-e offers a UART-interface to communicate with a host using a user-defined baud rate of up to 460800 Baud.

Additional Local/Remote GPIOs: The Thyone-e firmware allows configuration and control of free digital I/O pins on the module via serial or radio interface. More information can be found in chapter 10.

Transparent mode: A transparent mode is available out-of-the-box, enabling easy serial cable replacement. More information can be found in chapter 8.

Network addressing: The Thyone-e implements network addressing to enable unicast, multi-cast as well as broadcast data transmission. Additionally, packet ACK is available with automatic retry mechanism, to ensure reliable data transmission.

Mesh network: The Thyone-e offers repeater functionality to enable the creation of a simple flooding mesh network. The repeater mode can also be used for range extension.

1.2. Block diagram

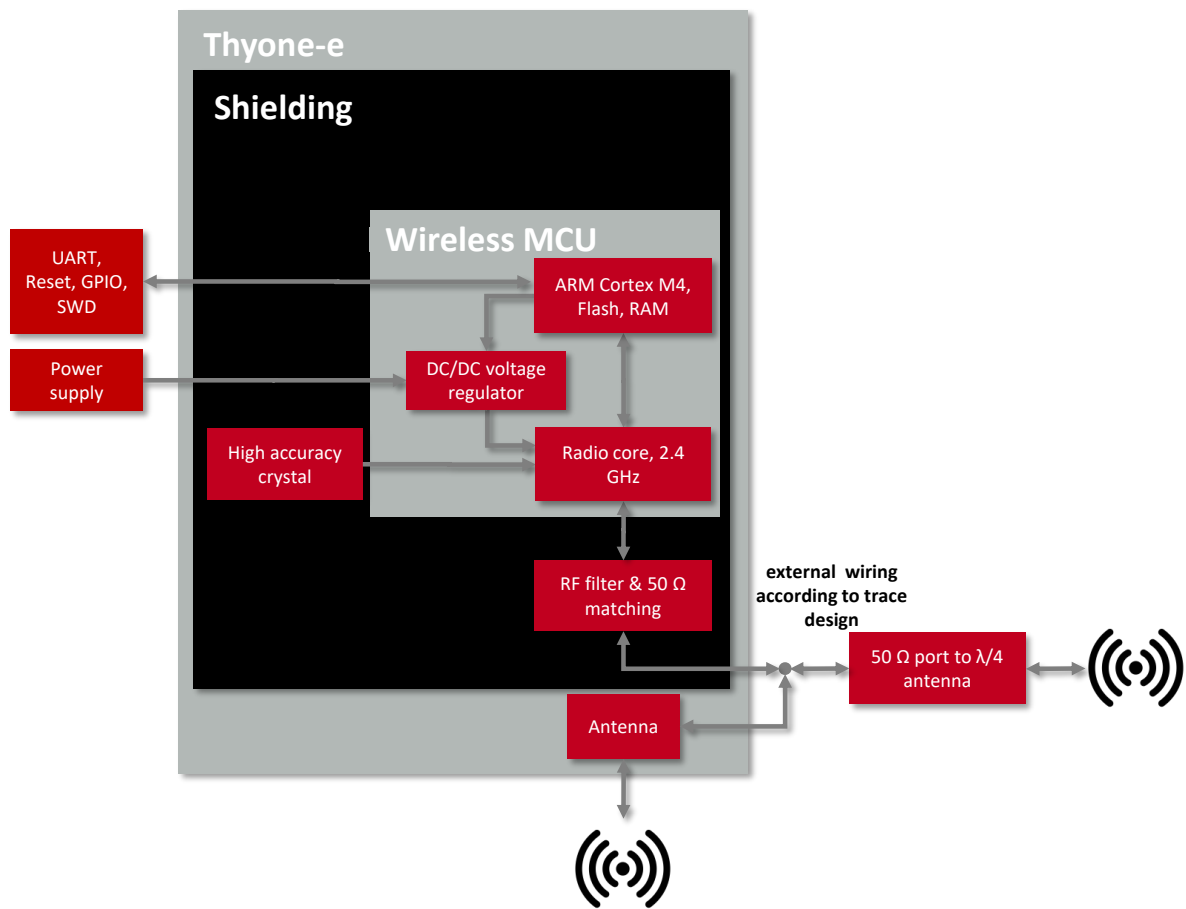


Figure 2: Block diagram of the module

1.3. Ordering information

| WE order code | Description |
|---------------|---|
| 2612011021000 | Thyone-e Module, Tape & Reel |
| 2612019021001 | Evaluation Kit for Thyone-e with 2 Thyone-e evaluation boards and accessories |

Table 1: Ordering information

2. Electrical specifications

Unless otherwise stated, the values specified are measured on the evaluation board, Thyone-e-EV with $T = 25\text{ °C}$, $VDD = 3\text{ V}$, $f = 2.44\text{ GHz}$, internal DC-DC converter in use.

2.1. Recommended operating conditions

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

Table 2: Recommended operating conditions



The on-chip power-on reset circuitry may not function properly for rise times longer than the specified maximum. A step 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 unstable supply voltage may significantly decrease the radio performance and stability.

2.2. Absolute maximum ratings

| Description | Min. | Typ. | Max. | Unit |
|--|-------|------|-------------|--------------------|
| Supply voltage (VDD) | -0.3 | | +3.9 | V |
| Voltage on any digital pin, $VDD < 3.6\text{V}$ | -0.3 | | $VDD + 0.3$ | V |
| Voltage on any digital pin, $VDD \geq 3.6\text{V}$ | -0.3 | | 3.9 | V |
| Input RF level | | | 10 | dBm |
| Flash endurance | 10000 | | | Write/erase cycles |

Table 3: Absolute maximum ratings

¹Power fail comparator is set to 1.8V to avoid flash fail due to voltage drop.

2.3. Power consumption

| Parameter | Power | Test conditions | Typ. | Unit |
|------------------------|-----------------|---|------|------|
| TX Current consumption | RF_TX_POWER = 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_TX_POWER = 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 4: 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 5: Current consumption - receiving

| Parameter | Test conditions | Typ. | Unit |
|---------------------|-------------------------|------|------|
| Current consumption | Sleep (system off mode) | 0.3 | µA |

Table 6: Current consumption - low power



Sleep current increases significantly for temperatures above 30 °C.

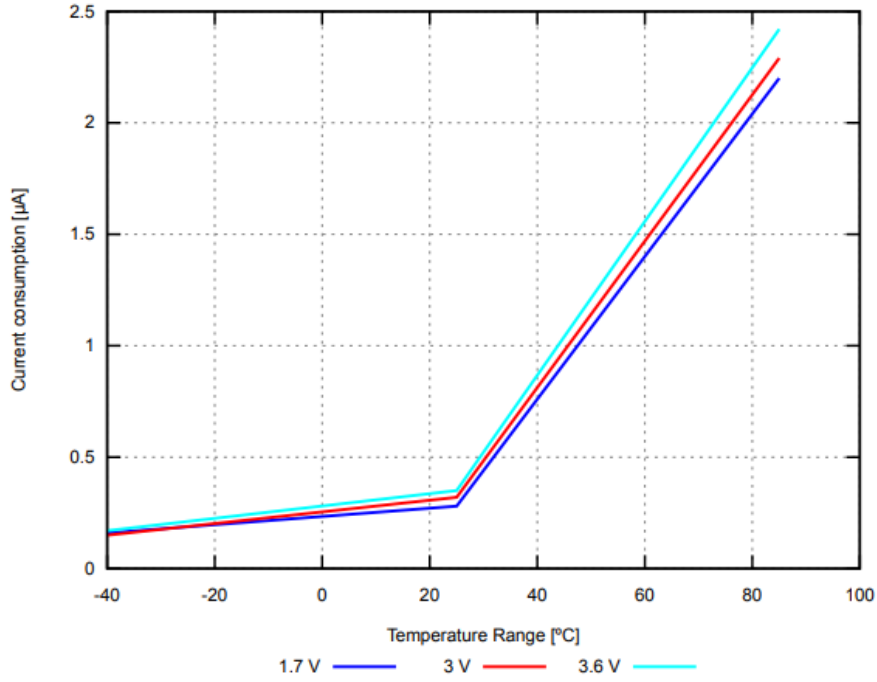


Figure 3: Sleep current (no RAM retention, wake on reset) over operating temperature range

2.4. Radio characteristics

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

Table 7: 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 8: Timing

| Parameter | Test conditions | Typ. | Unit |
|-------------------|--------------------------------------|------|------|
| Output power | RF_TX_POWER = 4, conducted | +4 | dBm |
| | RF_TX_POWER = 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 9: Transmit and receive power

All transmit and receive power levels are measured on the evaluation 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 bytes, at different data rates. The PER 1% is a harder criteria, resulting in 2 dBm less sensitivity.

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

Table 10: Sensitivity at different data rates

2.5. Pin characteristics

Specifications from nRF52 data sheet are reported here below.

| 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 11: Pin characteristics

3. Pinout

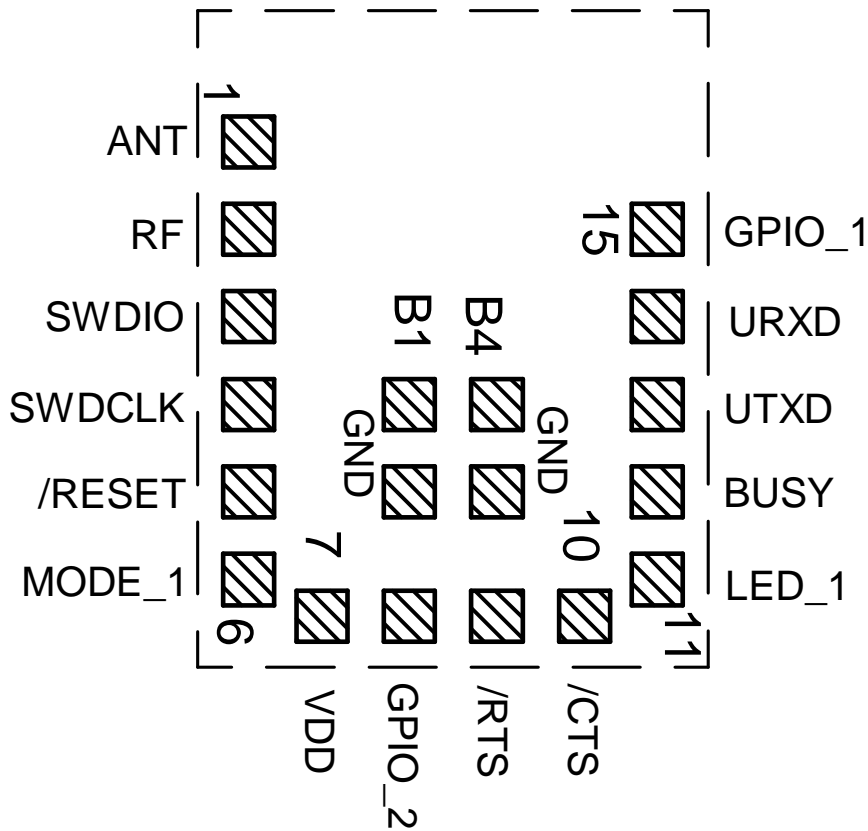


Figure 4: Pinout (top view)

| No | µC Pin | Designation | I/O | Description |
|----|--------|---------------|-------|---|
| 1 | | <i>ANT</i> | I/O | 50 Ω RF connection to PCB antenna. (see chapter 4.2). |
| 2 | | <i>RF</i> | I/O | 50 Ω RF connection through radio front end to transceiver part of chipset. (see chapter 4.2). |
| 3 | | <i>SWDIO</i> | I/O | 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> | I/O | Pin for remote GPIO access. Do not connect, if not needed. |
| 9 | P0.04 | <i>/RTS</i> | Output | <i>/RTS</i> signal, if flow control is enabled. Static LOW, otherwise. Do not connect if not needed. |
| 10 | P0.14 | <i>/CTS</i> | Input | <i>/CTS</i> signal, if flow control is enabled. Using internal pull down ¹ , otherwise. 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</i> | Output | This pin indicates if the module is busy with data transmission. Do not connect if not needed. |
| 13 | P0.16 | <i>UTXD</i> | Output | UART (Transmission). |
| 14 | P0.18 | <i>URXD</i> | Input | UART (Reception). Uses internal pull up resistor ¹ . |
| 15 | P0.20 | <i>GPIO_1</i> | I/O | 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 12: 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 Thyone-e does not implement this function.

4. Quick start

The Thyone-e module comes pre-flashed, tested and ready to use out-of-the box. Additionally, the module is compliant with the several regulatory requirements, which makes it suitable for use in most parts of the world (see chapter 22). This chapter describes steps to quickly build a prototype system and test the capabilities of the module.

4.1. Minimal pin connections

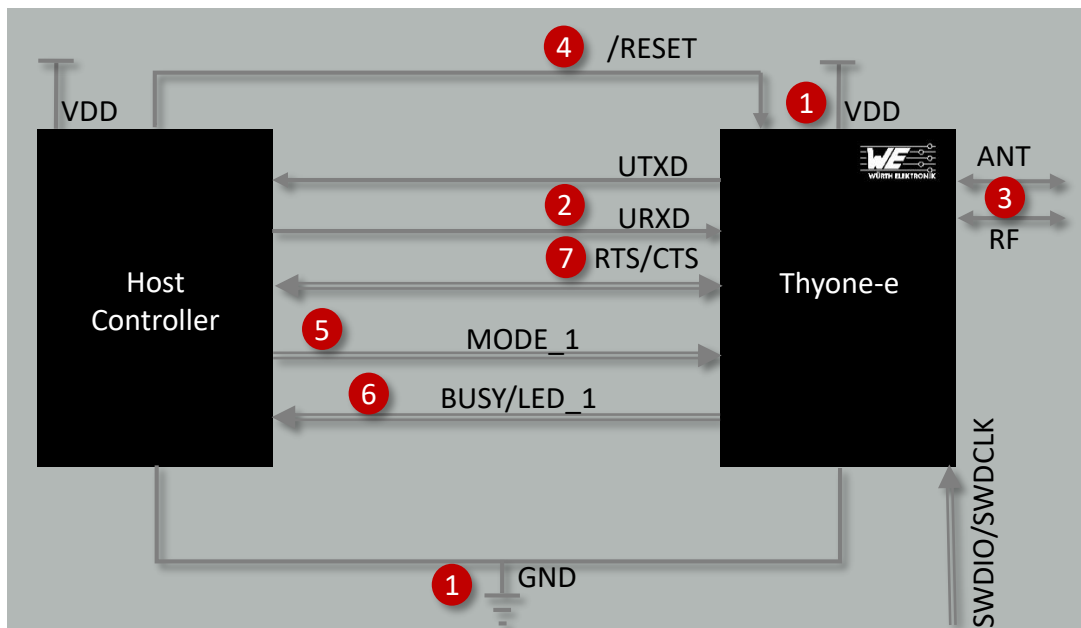


Figure 5: Minimal pin connections

The above image shows the bare minimum steps to be performed to integrate the Thyone-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 *UART RX* and *UART TX* pins of the host controller. Both UART lines are also required to be used for the firmware update procedure.

3. Antenna connection

Configure the antenna as per the requirement of the application. See section 4.2.

4. Reset

Connect the */RESET* pin to the host to allow a hard reset of the module.

5. (Optional) Mode selection

Connect the *MODE_1* pin to a digital output pin of the host controller to switch between command and transparent mode.

6. (Optional) Status indication

Connect the *BUSY* and *LED_1* pins to digital input pins of the host controller to allow easy status indication.

7. (Optional) UART flow control

In case baud rates higher than 115200 Baud are used, the UART flow control is activated automatically. In this case, the */RTS* and */CTS* pins must be connected to the */CTS* and */RTS* pins of the host controller.

The firmware update procedure requires the UART flow control lines to be connected to the host that performs the update.

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 3 for details on all pins. Furthermore, refer to chapter 16 for a reference design.



Implementing the firmware update connection in the customer's PCB is highly recommended.

For a secure firmware update via the UART interface, the module pins 13 to 16 (*URXD*, *UTXD* as well as */RTS* and */CTS*) shall be accessible. This allows the connection to a host PC for a UART firmware update. It is strongly recommended to consider this in the PCB layout of the final application.



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

4.2. Antenna connection

Thyone-e's smart antenna configuration enables the user to choose between two antenna options.

4.2.1. On-board PCB antenna

The Thyone-e has an on-board PCB antenna optimized for operating in the 2.4 GHz frequency band. A simple short between the pins *RF* and *ANT* feeds the RF output of the module to the on-board antenna of the Thyone-e. In this configuration, the module does not require any additional RF circuitry. For US and Canada, refer to the trace design in chapter 16.4.



The integrated antenna is ideal for compact designs where miniaturization is a priority. This configuration also enables re-use of module certifications for the end-application.

4.2.2. External antenna

For applications that use an external antenna, the Thyone-e provides a 50 Ω RF signal on pin *RF* of the module. In this configuration, pin *ANT* of the module has to be connected to ground and pin *RF* to the external antenna via 50 Ω feed line. Refer to chapter 16 for further information.



The external antenna configuration enables optimization of radio range at the cost of additional space on the PCB. This configuration is ideal in situations where a specialized antenna is necessary. For example, when metal housings are used.

4.3. Power up

After powering up the module, the */RESET* pin shall be held LOW for another Δt of 1 ms after the *VDD* is stable, to ensure a safe start-up. In the command mode, the module sends a *CMD_START_IND* on the */UTXD* to indicated "ready for operation". This indication is done with *BUSY* pin pulled to LOW in the transparent mode. See section 12.1 for more details regarding the start-up sequence.

4.4. Quick start example

This section is intended to help the user set-up and test the exchange data between two Thyone-e modules. Minimal pin and antenna connections have to be done on both the modules, as described in sections 4.1 and 4.2. It is recommended to use the Thyone-e EV-Kit for quick tests.

4.4.1. Prerequisites

The following hardware is required to go through the quick start example.

1. Two Thyone-e EV-Boards
2. Computer with WE UART Terminal PC tool or any other a serial terminal emulator.

4.4.2. Hardware configuration

The Thyone-e EV-Board can be used in its default configuration to go through this example. Refer to the Thyone-e EV-Board specific manual for a complete hardware description.

4.4.3. Set-up description

In this example, the Thyone-e EV-Boards are connected to the PC with WE UART Terminal tool installed (Figure 6).

4.4.4. Start-up

1. Connect the Thyone-e EV-Boards to the laptop/PC via USB.
2. The power LED indicates that supply voltage is active.



The FTDI driver [1] for the converter IC on the EV-Board has to be installed and/or updated. On correct driver installation, the EV-Board appears as a virtual COM port.

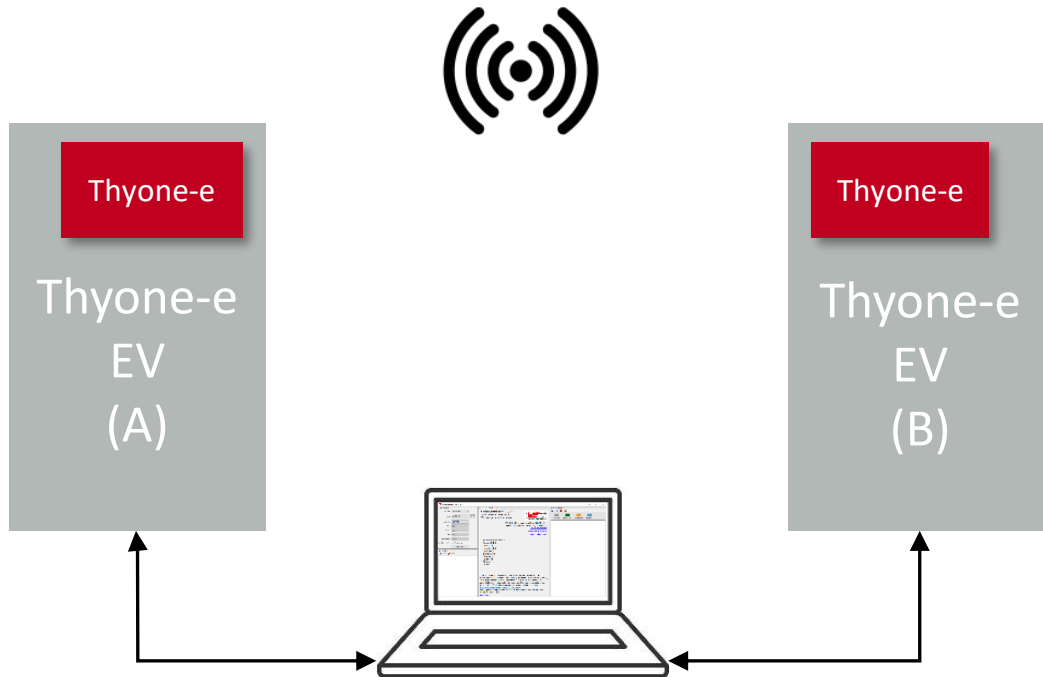


Figure 6: Set-up quick start


3. Open two instances of the WE UART Terminal [2].
4. Open an instance of the serial port with COM port settings 115200 Baud, 8n1 for each Thyone-e EV-Board in the WE UART Terminal instances.
5. By pressing the Reset button, the `CMD_START_IND` appears on the command window of the WE UART Terminal.

| Info | Module A | Module B |
|--|----------------------------|----------------------------|
| ⇐ Response <code>CMD_START_IND</code> : Module A started in command mode. Reset reason: Reset pin | 02 73 03 00 00 02 01 71 | |
| ⇐ Response <code>CMD_START_IND</code> : Module B started in command mode. Reset reason: Reset pin | | 02 73 03 00 00 02 01 71 |

4.4.5. Transmit and receive data

1. On the WE UART Terminal connected to Thyone-e-EV A, enter hex-equivalent of "hello world", "0x68 0x65 0x6C 0x6C 0x6F 0x20 0x77 0x6F 0x72 0x6C 0x64" in the data transmission text box.
2. Click the CMD_BROADCAST_DATA_REQ button to trigger transmission.
3. The module acknowledges with a CMD_DATA_CNF and sends a CMD_TXCOMPLETE_RSP indicating successful transmission of data.
4. On the WE UART Terminal connected to Thyone-e-EV B, a CMD_DATA_IND message appears containing the same data that was transmitted.
5. Repeat the above steps from Thyone-e-EV B to test data transmission from B to A.
6. Note that the TX and RX LEDs blink to indicate data transmission and reception, respectively.

| Info | Module A | Module B |
|---|---|---|
| ⇒ Request CMD_BROADCAST_DATA_REQ: Send "hello world" | 02 06 0B 00 68 65 6C 6C 6F 20 77 6F 72 6C 64 2F | |
| ⇐ Response CMD_DATA_CNF: Request received, sending data now. | 02 44 01 00 00 47 | |
| ⇐ Response CMD_TXCOMPLETE_RSP: Data transmitted successfully. | 02 C4 01 00 00 C7 | |
| ⇐ Indication CMD_DATA_IND: Received hello world from a module with address 6C000001 with RSSI 0xC8(-56 dBm) | | 02 84 10 00 01 00 00 6C C8 68 65 6C 6C 6F 20 77 6F 72 6C 64 13 |
| ⇒ Request CMD_BROADCAST_DATA_REQ: Send "hello world" | | 02 06 0B 00 68 65 6C 6C 6F 20 77 6F 72 6C 64 2F |
| ⇐ Response CMD_DATA_CNF: Request received, sending data now. | | 02 44 01 00 00 47 |
| ⇐ Response CMD_TXCOMPLETE_RSP: Data transmitted successfully. | | 02 C4 01 00 00 C7 |
| ⇐ Indication CMD_DATA_IND: Received hello world from a module with address 6C000002 with RSSI 0xC7(-57 dBm) | 02 84 10 00 02 00 00 6C C7 68 65 6C 6C 6F 20 77 6F 72 6C 64 1F | |

 The RSSI values will differ based on the distance between the modules.

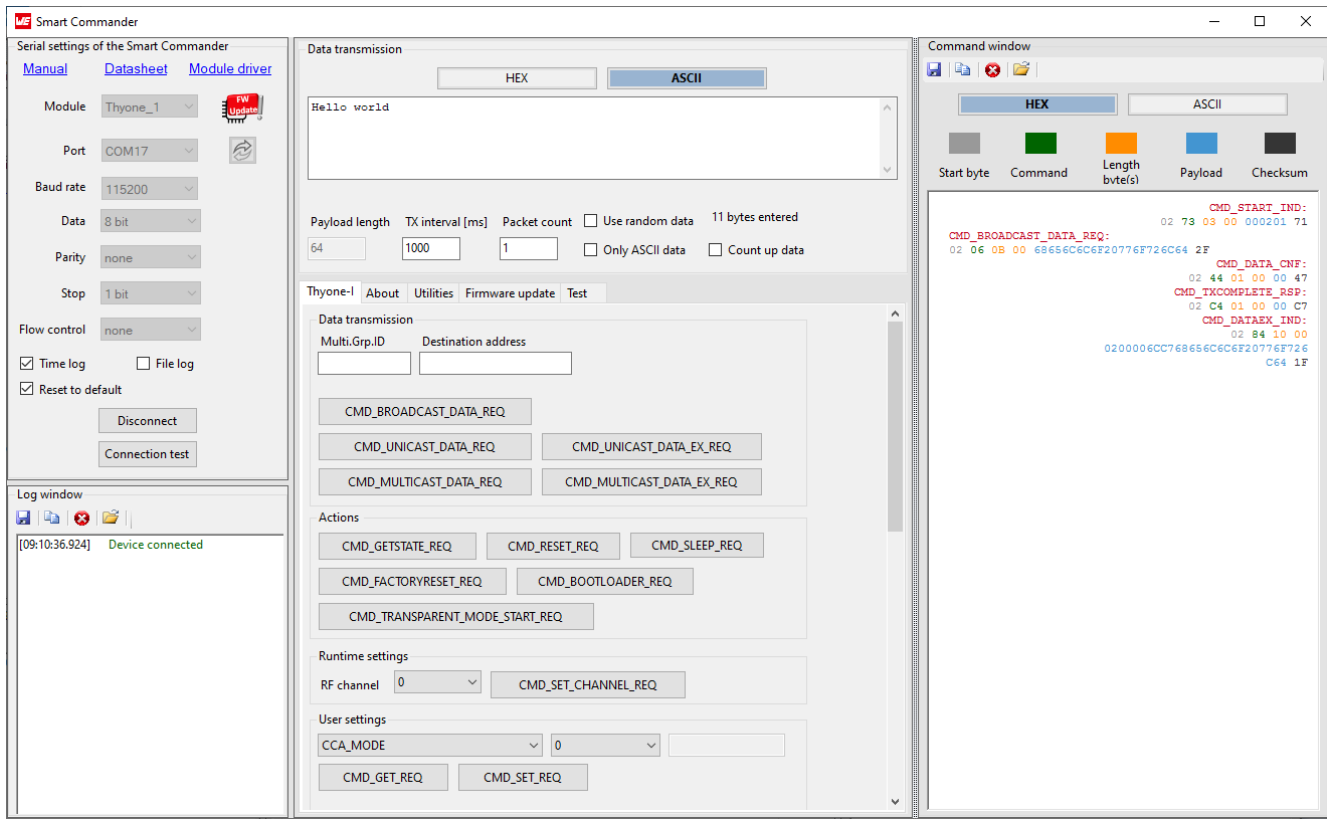


Figure 7: Module A command sequence

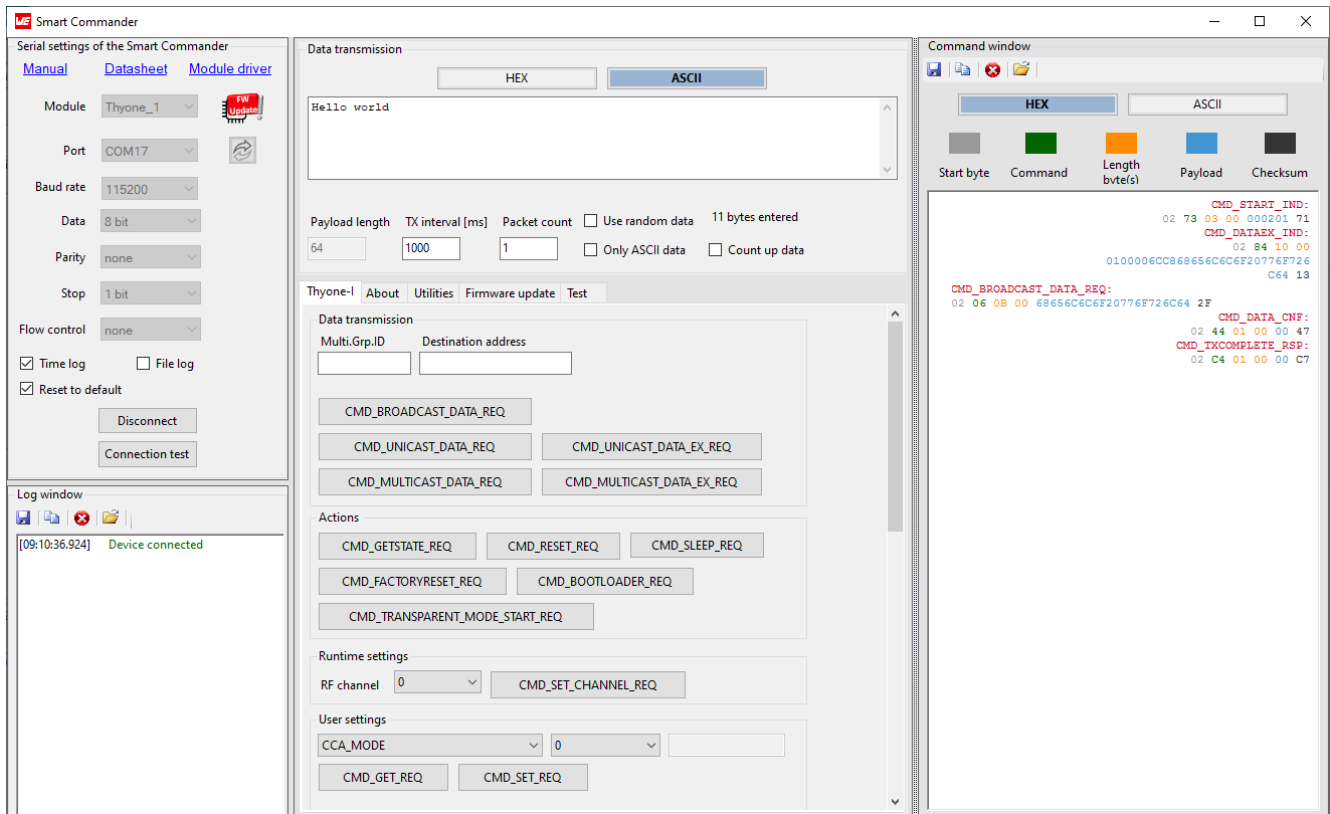


Figure 8: Module B command sequence

5. Functional description

The Thyone-e module is a radio sub-module operating in the globally available 2.4 GHz license free band. Interfacing with the host system via serial UART, the module allows easy configuration and control of the radio using a simple command interface. The module offers a range of features including:

- High throughput mode
- Low energy consumption
- Flexible network addressing
- Point-to-point communication with ACK and retries
- Flooding mesh network using the repeater function
- Encrypted radio communication
- Local/Remote digital I/O control
- Transparent mode
- Test modes for radio conformity/certification tests.

In this chapter, a detailed description of various functional features of the module is presented.

5.1. Radio

Thyone-e uses 40 different 1 MHz channels in the license free 2.4 GHz band (2402-2480 MHz). The data is GFSK modulated.

5.1.1. Radio profiles

The radio profile determines the modulation and coding scheme used in the physical layer. The choice of the radio profile directly influences the achievable range and throughput.

Thyone-e can be configured to use one of the radio profiles listed in table 13. All radio profiles support a packet size of maximum 224 bytes.

| Radio profile | Data rate (gross) [kbit/s] | Throughput (end-to-end) ¹ [kbit/s] | Range ² [m] |
|---------------|----------------------------|---|------------------------|
| 2 | 1000 | 239 | 188 |
| 3 | 2000 | 275 | 158 |

Table 13: Radio profiles

¹Maximum rate of transfer of data from host micro-controller on the transmit side over UART to the module and further via radio link and UART to the host micro-controller on the receive side.

²Calculated using the two way ground reflection model with a transmission power of 4 dBm, external antenna placed 2 m above the ground, 1% PER (packet error rate), 0 dB antenna gain and a reserve of 6 dB.

5.1.2. RF channel

Thyone-e operates in the 2.4 GHz band and can be configured to use one of the 40 channels from 2402 MHz to 2480 MHz.

- The bandwidth of each of the channels is 1 MHz.
- The distance between two adjacent channels is 2 MHz.
- The parameter `RF_CHANNEL` can be used to set the default channel for radio transmission and reception.
- The command `CMD_SETCHANNEL_REQ` (section 7.5.7) can be used to change the channel in runtime. This setting is however lost after a reset.

5.1.3. RF transmit power

The transmit power of the Thyone-e module can be configured from -40 dBm to +4 dBm using the parameter `RF_TX_POWER`. Transmit power has a direct influence on the achievable radio range and the current consumption of the module.

5.1.4. Radio compatibility to Thyone-I

The Thyone-e and Thyone-I radio modules are not radio compatible¹. The main reason is that the radio channels (frequencies) of both products do not coincide.

Furthermore, the Thyone-e does not support the default profile of Thyone-I, which is profile 0.

5.2. Low power mode

The Thyone-e module is optimized to minimize the energy consumption, in order to be used in a battery-driven application. A sleep mode is available, in which the module draws a current less than 1 μ A. The module can be sent to sleep using the command `CMD_SLEEP_REQ`. In this mode, the UART is disabled and the module does not transmit or receive any data over the radio. In order to prevent any leakage current, the host has to pull the `URXD` to LOW level. The GPIOs `GPIO_1` and `GPIO_2` are set to input during the sleep period.

To leave this state, a pin reset must be applied. The module restarts and all the volatile settings are lost.

5.3. Network addressing

In order to build a network on interconnected devices and send data to specific devices, an addressing mechanism is necessary. Thyone-e implements addressing mechanism to support unicast, multicast and broadcast transmission of data.

- Every module can be configured to have a unique 4-byte address. By default, the source address of the module is configured to be the serial number of the module.

¹A radio compatible version of the Thyone-e can be provided under request. In such case, contact your BDM or WCS@we-online.com for further information.

- Additionally, the module can be configured to transmit to a group using a 1-byte group-ID. This feature enables creation of multicast groups.
- Data commands are available to send unicast/multicast or broadcast data (see section 7.3).
- Automatic address resolution is implemented to filter out the packets that are not addressed to the module's configured address.
- Network addressing is also available in the transparent mode (see chapter 8).
- The parameters `MAC_SOURCE_ADDRESS`, `MAC_GROUP_ID`, `MAC_DEST_ADDRESS` and `MAC_TRANSPARENT_ADDR_MODE` can be used to set the network settings. See chapter 9 for further details.

5.4. Acknowledgment and retries

In order to improve reliability in communication, the module can be configured to use the radio acknowledgment and retry mechanism. It can be activated setting the parameter `RF_NUM_RETRIES` accordingly. Retries are not performed for broadcast messages.

5.5. Repeater mode

Thyone-e can be run as repeater to extend the transmission range. A module configured as a repeater, simply retransmits the received packet after a random back-off time. A time-to-live (TTL) parameter (`MAC_TTL`) can be used to set the hop limit. This mode allows building a flooding mesh network, as described in chapter 11.

5.6. Packet sniffer mode

The module can be configured to be used as a sniffer by setting the parameter `MODULE_MODE`. In this mode, the address resolution is disabled and the module accepts all the packets (irrespective of the target address) and forwards it over the UART. In this mode, the ACKs are disabled. This mode cannot be used along with the repeater mode. If a sniffer receives an encrypted packet, it forwards the packet only when the packet can be decrypted (the correct key is present).

5.7. Encryption/decryption

In order to establish a secure wireless link, the Thyone-e supports encryption and decryption of transmitted payload.

- The Thyone-e implements a symmetric 128 bit AES encryption.
- Encryption is disabled by default.
- Encryption/decryption of data cannot be enabled without setting the `MAC_ENCRYPTION_KEY`.
- The 128 bit key can be written on to the flash using the parameter `MAC_ENCRYPTION_KEY`.

- The parameter `ENCRYPTION_MODE` can be used to configure the level of security desired (see table 14).
- See chapter 9 for details on the configuration of the encryption parameters.

| ENCRYPTION_MODE | Transmission | Reception |
|-----------------|--------------|--|
| 0x00 | Unencrypted | Unencrypted and encrypted packets (with matching key) |
| 0x01 | Encrypted | Unencrypted and encrypted packets (with matching key) |
| 0x02 | Unencrypted | Encrypted packets only (unencrypted and non-decryptable packets are discarded) |
| 0x03 | Encrypted | Encrypted packets only (unencrypted and non-decryptable packets are discarded) |

Table 14: Security levels



Security warning: The 128 bit AES key is write-only and cannot be read over UART. However, physical access to the flash may enable malicious users to access the key. Hence, leading to a potential vulnerability. It is recommended to use secure storage for user keys in applications with high security requirements.

5.8. Local/remote digital I/O control

Thyone-e offers several programmable digital I/O pins for use in end applications. These pins can be configured and controlled directly by the local host over UART or by a remote host over the radio link. Chapter 10 explains the options available to configure and control the digital I/O pins in detail.

5.9. Transparent mode

Often, legacy systems using a serial cable for communications need to be upgraded to use a wireless link. In order to support such applications and other applications where simplicity is preferred over versatility, Thyone-e offers the transparent mode. In this mode, the data sent over the UART is not interpreted by the module (as command). It is blindly forwarded over the radio link. Similarly, all data received over the radio link are forwarded over the UART. For details regarding the configuration of transparent mode, refer to chapter 8.

5.10. Radio test mode

In order to test the RF performance of the module after integration into a custom design, the Thyone-e offers radio test modes. Thyone-e implements essential test features out-of-the-box.

- Transmit carrier on a specific channel at a specific transmit power.
- Transmit a random or predefined packet on a specific channel with a given transmit power.
- Receive on a specific channel

Test features come in handy when performing radio compliance tests. Section 7.8 describes the test mode commands in detail.



This mode is intended for design verification and compliance tests only. It should not be used under normal operation.

5.11. Modes of operation

When powered on, the Thyone-e can be in one of the following modes of operations:

- **Reset** A state where the */RESET* is held LOW.
- **Command mode** This is the standard mode of operation for Thyone-e. In this mode, the module can be configured and controlled using the command interface (see chapter 7).
- **Transparent mode** In this mode, the module acts as a transparent UART-radio bridge (see chapter 8).
- **Radio test mode** This is a mode meant for radio performance and compliance testing (see chapter 7.8).
- **Sleep** The state of lowest power consumption where the module is waiting for a wake-up trigger.

Figure 9 illustrates various modes of operation and transitions to/from each of the above states.

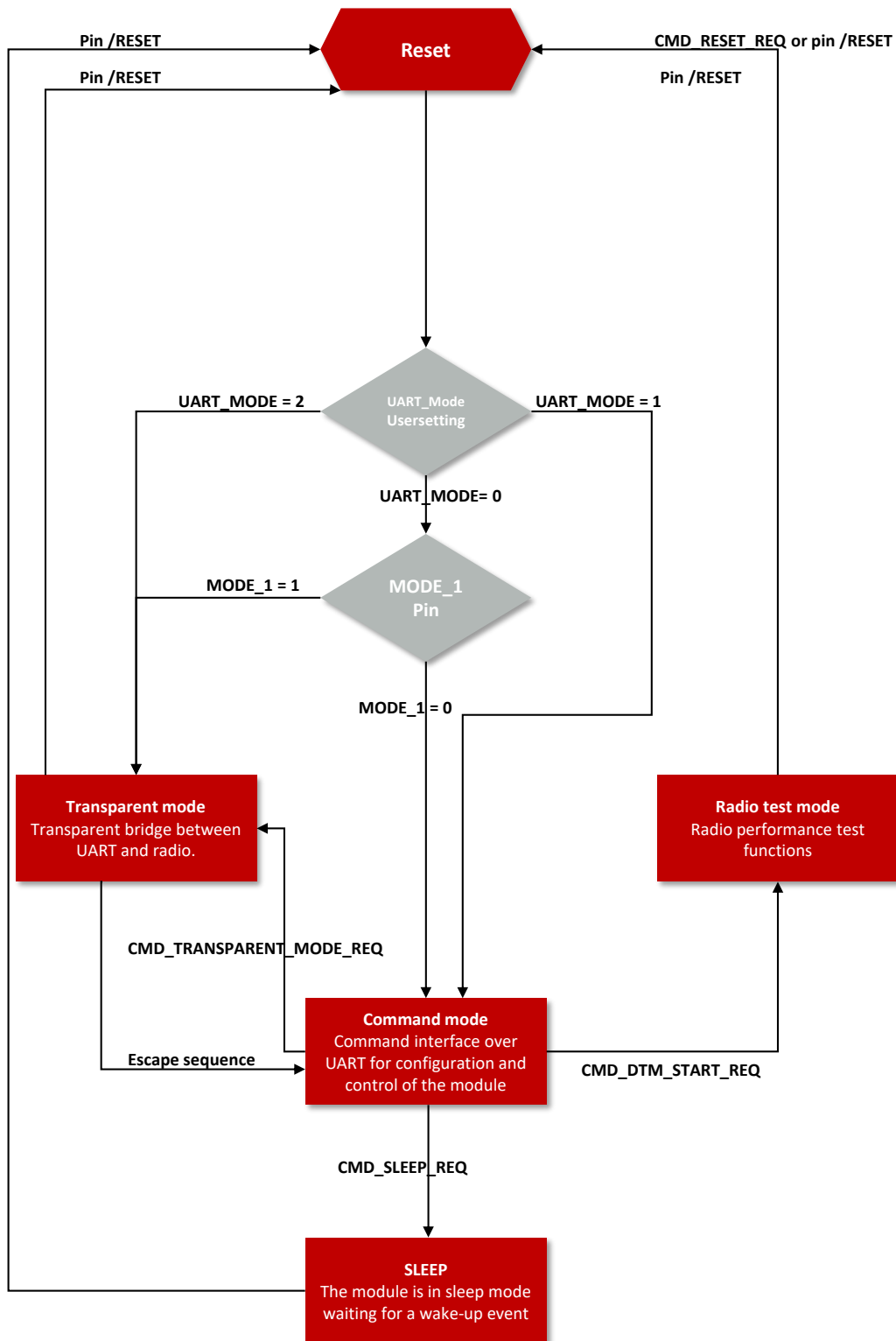


Figure 9: Overview modes

6. Host connection

The Thyone-e is intended to be used as a radio module in a system interfaced with a host micro-controller. The use of an industry standard UART as the primary interface, ensures a very minimal requirement set on the host MCU. As a result of this, the module can be designed in with most host controllers: from an 8051 to the more advanced ARM core architectures.

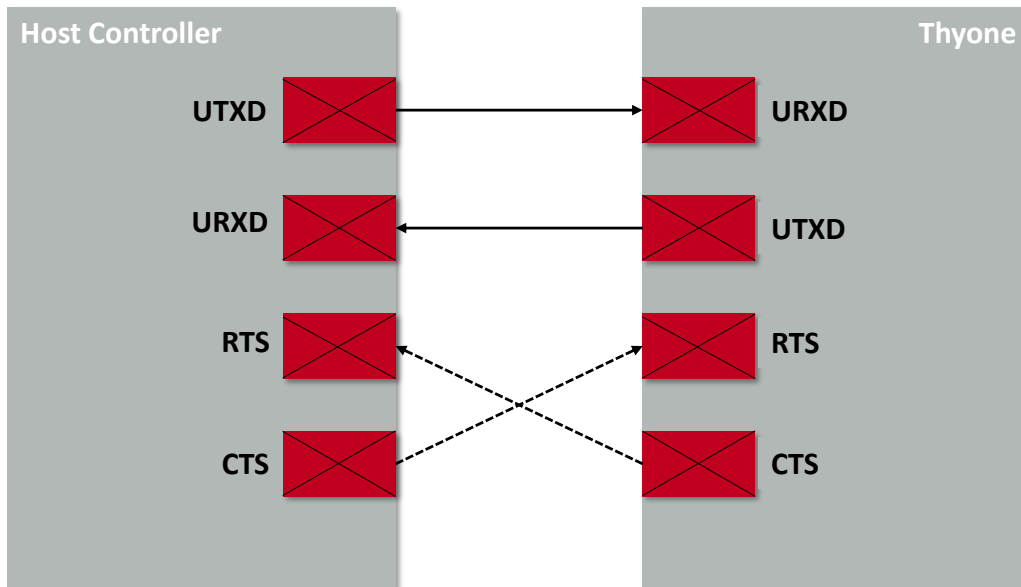


Figure 10: Host interface

6.1. Serial interface: UART

The Thyone-e implements the standard UART interface with the following parameters.

| Parameter | Range | Standard |
|--------------|----------------|----------|
| Baud | 1200 to 460800 | 115200 |
| Data bits | 8 | 8 |
| Stop bits | 1 | 1 |
| Parity | none, even | none |
| Flow control | none, RTS/CTS | none |

Table 15: UART parameters

The baud rate, parity and flow control of the UART can be configured by means of the user setting `UART_CONFIG`.

6.1.1. Timing and reset behaviour

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.

When holding the module's */RESET* pin LOW, the radio chip states are undefined. In this case, the module's *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 may not accept bytes with frame errors and flush its RX buffer, after pulling the module's */RESET* pin LOW.

7. The command interface

The Thyone-e is interfaced with a host micro-controller using the serial UART interface. The configuration as well as the operation of the module can be managed by predefined commands that are sent as packets over this UART interface.

7.1. Command categories

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

- **Requests:** The host requests the module to trigger an action, e.g. the request `CMD_RESET_REQ` 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`. Every response message contains a status byte indicating status of the issued request. Table 16 lists the status byte values and corresponding status messages.
- **Indications and Responses:** The module indicates spontaneously when a special event has occurred. For example, the `CMD_DATA_IND` indicates that a packet was received over the radio.

7.2. Command structure

The structure of the command request/indicate packet is as shown below.

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

The structure of the command confirmation/response packet is as shown below.

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

Start signal: 0x02 (1 Byte)

Command: One of the predefined commands (1 Byte).

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

Status: Present only in Confirmations and Responses. Indicates the status of previous request message. (see table 16)

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

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

| Status byte | Description |
|-------------|---|
| 0x00 | Requested action success |
| 0x01 | Requested action failed ("Parameter(s) out of bound", "Invalid request packet" or "Ack not received") |
| 0x02 | Module is busy with an on-going RF operation. Request cannot be processed |
| 0x03 | Error occurred while triggering the requested operation |
| 0x04 | Key not found |
| 0x05 | Supply voltage below 1.8 V ($\pm 5\%$) |
| 0xFF | Operation not permitted in the current state |

Table 16: Confirmation status byte



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 [3].



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 will discard the received bytes and wait for a new command. This means that the delay between two successive bytes in a frame must be kept as low as possible.

7.3. User data commands

In this section, the commands available to transmit and receive user data are described in detail.

7.3.1. CMD_BROADCAST_DATA_REQ

This command provides the simple broadcast data transmission. A payload length of maximum 224 bytes can be transmitted per packet.

When the data is processed by the module a CMD_DATA_CNF is output on the UART. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been transmitted over the air.

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

Format:

| Start signal | Command | Length | Payload | CS |
|--------------|---------|---------|--------------|--------|
| 0x02 | 0x06 | 2 Bytes | Length Bytes | 1 Byte |

7.3.2. CMD_MULTICAST_DATA_REQ

This command provides the multicast data transmission to a group of modules configured with the same MAC_GROUP_ID. The module uses the default multicast group ID stored in the user settings parameter, MAC_GROUP_ID. A payload length of maximum 224 bytes can be transmitted per packet.

When the data is processed by the module a CMD_DATA_CNF is output on the UART. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been transmitted over the air.

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

Format:

| Start signal | Command | Length | Payload | CS |
|--------------|---------|---------|--------------|--------|
| 0x02 | 0x05 | 2 Bytes | Length Bytes | 1 Byte |

7.3.3. CMD_UNICAST_DATA_REQ

This command can be used for unicast data transmission. The module sends data to the default destination address stored in the user settings parameter MAC_DEST_ADDRESS. A payload length of maximum 224 bytes can be transmitted per packet.

When the data is processed by the module a CMD_DATA_CNF is output on the UART. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been transmitted over the air.

The receiving Thyone-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 |

7.3.4. CMD_MULTICAST_DATA_EX_REQ

This command can be used to multicast data to a group ID specified in the packet. A payload length of maximum 224 bytes can be transmitted per packet.

When the data is processed by the module a CMD_DATA_CNF is output on the UART. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been transmitted over the air.

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

Format:

| Start signal | Command | Length | Grp ID | Payload | CS |
|--------------|---------|---------|--------|--------------------|--------|
| 0x02 | 0x08 | 2 Bytes | 1 Byte | (Length - 1) Bytes | 1 Byte |

7.3.5. CMD_UNICAST_DATA_EX_REQ

This command can be used to transmit data to the destination address sent along with the packet. The destination address in the message is in LSB first format. A payload length of maximum 224 bytes can be transmitted per packet.

When the data is processed by the module a CMD_DATA_CNF is output on the UART. Additionally a CMD_TXCOMPLETE_RSP will follow as soon as the data has been transmitted over the air.

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

Format:

| Start signal | Command | Length | Dest Addr | Payload | CS |
|--------------|---------|---------|-----------|--------------------|--------|
| 0x02 | 0x07 | 2 Bytes | 4 Bytes | (Length - 4) Bytes | 1 Byte |

7.3.6. CMD_DATA_CNF

This message is sent by the module in response to any of the data transmit request messages.

Format:

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

7.3.7. CMD_TXCOMPLETE_RSP

This command is output on the UART as soon as the data transmission is complete.

Format:

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

In case of broadcast or multicast messages the Status byte is 0x00, which indicates that the message has been transmitted successfully.

For unicast messages, the radio module waits for the acknowledgement of the receiver, and thus outputs 0x00 (in case of successful data reception) or 0x01 (in case of no acknowledgement) as Status byte.

7.3.8. CMD_DATA_IND

This message indicates the reception of a valid data packet on the radio. The CMD_DATA_IND returns the MAC_SOURCE_ADDRESS of the sending device, the RSSI value of the received data packet and the data received via the RF-interface. The RSSI value is output in two's complement notation.

Format:

| Start signal | Command | Length | Src Addr | RSSI | Payload | CS |
|--------------|---------|---------|----------|--------|--------------------|--------|
| 0x02 | 0x84 | 2 Bytes | 4 Bytes | 1 Byte | (Length - 5) Bytes | 1 Byte |

7.3.9. CMD_SNIFFER_IND

This message indicates the reception of a valid data packet on the radio in sniffer mode. The CMD_SNIFFER_IND returns the MAC_SOURCE_ADDRESS of the sending device, the RSSI value of the received data packet and the data received via the RF-interface. The RSSI value is output in two's complement notation.

Format:

| Start signal | Command | Length | Src Addr | RSSI | DATA_IND | Payload | CS |
|--------------|---------|---------|----------|--------|----------|--------------------|--------|
| 0x02 | 0x99 | 2 Bytes | 4 Bytes | 1 Byte | 0x84 | (Length - 6) Bytes | 1 Byte |

7.4. Configuring the module

The module's parameters, also called "User settings", are stored in flash. A local copy of the user settings also known as "Runtime settings" are stored in the RAM. The flash parameters can be modified by the `CMD_SET_REQ`, read by the `CMD_GET_REQ`. The values stored in the flash are retained even after a power cycle.

It is possible to get modules pre-flashed with custom settings out-of-the-box. This prevents in-field configuration and enables direct use in the desired mode of operation. See Chapter 13 for details regarding firmware individualization.

7.4.1. `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 23.

Parameters of 2 or more bytes have to be transferred (both to and from the module) 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 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 |

7.4.1.1. Example 1

Setting the RF channel to channel 10.

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|----|
| 0x02 | 0x11 | 0x02 0x00 | 0x07 | 0xA | 1C |

Response:

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

Setting was set successfully. The module restarts automatically after the confirm message.

7.4.2. 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 23.

Parameters of 2 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 |

7.4.2.1. Example 1

Request the serial number of the module.

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

Response:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---------------------|------|
| 0x02 | 0x50 | 0x05 0x00 | 0x00 | 0x01 0x00 0x00 0x6C | 0x3A |

Setting was read successfully.

7.5. Manage the device state

7.5.1. CMD_START_IND

This is an indication message that the module sends to the host micro-controller on boot-up in command mode. This message indicates that the module has successfully booted-up and is ready to receive commands from the host.

Format:

| Start signal | Command | Length | Status | Reset reason | Module mode | CS |
|--------------|---------|-----------|--------|--------------|-------------|--------|
| 0x02 | 0x73 | 0x03 0x00 | 1 Byte | 1 Byte | 1 Byte | 1 Byte |

Reset reason:

0x01: Power on or wake up from sleep

0x02: Pin reset

0x04: Soft reset

Others: Reserved

Module mode:

0x01: Application

0x05: Test mode

Other: Reserved

7.5.2. CMD_GETSTATE_REQ

This command returns the current state of the module.



Refer to chapter 5.11 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 | Length | Status | Module mode | CS |
|--------------|---------|-----------|--------|-------------|--------|
| 0x02 | 0x41 | 0x02 0x00 | 1 Byte | 1 Byte | 1 Byte |

Module mode:

0x01: Application

0x05: Test mode

Other: Reserved

7.5.3. 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 |

7.5.4. CMD_SLEEP_REQ

This command is used to start the system-off mode. After entering this mode, the module has to be woken up using the *WAKE_UP* pin (apply a LOW signal at this pin for at least 5 ms and release it to HIGH again) before any other action can be performed. The UART interface as well as the radio interface are shut down in this mode. The GPIOs *B1-B6* are set to input during the sleep period. For more details, refer to chapter 5.2.

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 |

7.5.5. CMD_FACTORY_RESET_REQ

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



Note that the GPIO configuration specified in chapter 10 is also reset to default.

Format:

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

Response (CMD_FACTORY_RESET_CNF):

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



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.

7.5.6. CMD_TRANSPARENT_MODE_REQ

This command restarts the module in transparent mode. In this mode, the module acts as a transparent bridge between UART and the radio interface. See chapter 8 for further details.

Format:

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

Response (CMD_TRANSPARENT_MODE_CNF):

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

7.5.7. CMD_SETCHANNEL_REQ

In order to change the radio channel on the fly, a CMD_SETCHANNEL_REQ can be used. This command modifies the volatile setting of the channel currently in use. The channel is set to the value in the user setting after a reset. Refer to table 21 for the frequency values corresponding to the channel index.

| Start signal | Command | Length | Channel Index | CS |
|--------------|---------|-----------|---------------|--------|
| 0x02 | 0x09 | 0x01 0x00 | 1 Byte | 1 Byte |

Response (CMD_SETCHANNEL_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x49 | 0x01 0x00 | 1 Byte | 1 Byte |

7.6. Digital I/O control

This chapter contains the commands to use the GPIO feature of the Thyone-e. Refer to chapter 10 for a detailed description.

7.6.1. CMD_GPIO_LOCAL_SET_CONFIG_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 write cycles. Try to avoid performing repeated CMD_GPIO_LOCAL_SET_CONFIG_REQ as each command will use one write cycle. When the configuration of the I/O pins is known in advance, an individualized firmware can be used to prevent in-field configuration (see chapter 13).

Format:

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

Response (CMD_GPIO_LOCAL_SET_CONFIG_CNF):

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

CMD_GPIO_LOCAL_SET_CONFIG_REQ block structure

Each **Block** has the following format:

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

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 10.1

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_SET_CONFIG_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 10.1

Status:

- 0x00:** Success
- 0x01:** Failed

7.6.1.1. 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.

7.6.2. CMD_GPIO_LOCAL_GET_CONFIG_REQ

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

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x26 | 0x00 0x00 | 0x24 |

Response (CMD_GPIO_LOCAL_GET_CONFIG_CNF):

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

CMD_GPIO_LOCAL_GET_CONFIG_CNF block structure

Each **Block** has the following format:

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

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 10.1

Function:

- 0x00:** GPIO disconnected
- 0x01:** GPIO works as input
- 0x02:** GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x02:
value 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

7.6.2.1. Example: Read the current GPIO configuration

Read the current configuration:

| Start signal | Command | Length | CS |
|--------------|---------|-----------|------|
| 0x02 | 0x26 | 0x00 0x00 | 0x24 |

Response:

| Start signal | Command 0x40 | Length | Status | Blocks | CS |
|--------------|----------------|-----------|--------|--|------|
| 0x02 | 0x66 | 0x15 0x00 | 0x00 | 0x03 0x01 0x02 0x01 0x03 0x02 0x02 0x01 0x02 0x03 0x00 0x02 0x04 0x00 0x02 0x05 0x00 0x02 0x06 0x00 | 0x76 |

The GPIOs with GPIO_ID **0x01** and **0x02** are output high. The remaining GPIOs with GPIO_ID **0x03**, **0x04**, **0x05** and **0x06** are disconnected.

7.6.3. CMD_GPIO_REMOTE_SET_CONFIG_REQ

This command configures the free GPIOs of the addressed 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 has to be addressed to a specific remote device with the 4-byte destination address.



The flash memory used to store these settings has a limited count of write cycles. Try to avoid performing periodic CMD_GPIO_REMOTE_SET_CONFIG_REQ as each command will use one write cycle. When the configuration of the I/O pins is known in advance, an individualized firmware can be used to prevent in-field configuration (see chapter 13).

Format:

| Start signal | Command | Length | Remote address | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------------------|-----|--------------------|--------|
| 0x02 | 0x29 | 2 Bytes | 4 Bytes | x Bytes | | x Bytes | 1 Byte |

Response from local Thyone-e module (CMD_GPIO_REMOTE_SET_CONFIG_CNF):

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

This response is followed by a CMD_TXCOMPLETE_RSP to indicate the completion of the packet transmission. If the packet was correctly received by the remote module, it responds with a confirmation message. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_SET_CONFIG_RSP:

| Start signal | Command | Length | Remote address | RSSI | Status | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xE9 | 2 Bytes | 4 Bytes | 1 Byte | 1 Byte | x Bytes | | x Bytes | 1 Byte |

CMD_GPIO_REMOTE_SET_CONFIG_REQ block structure

Each **Block** has the following format:

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

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 10.1

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 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

CMD_GPIO_REMOTE_SET_CONFIG_RSP block structure

Each **Block**, except Block 1, has the following format:

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



Block 1 is always 2 byte long containing only the GPIO_ID and the status byte. The leading length byte is not present.

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 10.1

Status:

- 0x00:** Success
- 0x01:** Failed
- 0xFF:** Remote configuration not allowed (blocked by the user setting GPIO_BLOCK_REMOTE_CONFIG of the remote device)

7.6.3.1. Example: Configure two GPIOs of the remote device to output HIGH

Configure the GPIOs with ID **0x01** and **0x02** of device with address 0x6C000002 to output HIGH:

| Start signal | Command | Length | Remote address | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|---------------------------|-------------------------------|-------------------------------|------|
| 0x02 | 0x29 | 0x0C 0x00 | 0x02 0x00 0x00 0x6C | 0x03 0x01 0x02 0x01 | 0x03 0x02 0x02 0x01 | 0x4A |

The local module responds with a CMD_GPIO_REMOTE_SET_CONFIG_CNF and CMD_TXCOMPLETE_RSP to indicate receipt of packet on the UART and transmission over radio respectively. On obtaining a response from the remote module, the host sends the following response CMD_GPIO_REMOTE_SET_CONFIG_RSP:

| Start signal | Command | Length | Remote address | RSSI | Status | Block ₁ | Block ₂ | CS |
|--------------|---------|---------|---------------------------|------|--------|---------------------|-----------------------------|------|
| 0x02 | 0xE9 | 2 Bytes | 0x02 0x00 0x00 0x6C | 0xC7 | 0x00 | 0x01 0x00 | 0x01 0x02 0x00 | 0x4D |

Configured both GPIOs with success.

7.6.4. CMD_GPIO_REMOTE_GET_CONFIG_REQ

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

Format:

| Start signal | Command | Length | Remote address | CS |
|--------------|---------|-----------|----------------|--------|
| 0x02 | 0x2A | 0x04 0x00 | 4 Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_GET_CONFIG_CNF):

| Start signal | Command 0x40 | Length | Status | CS |
|--------------|----------------|-----------|--------|--------|
| 0x02 | 0x6A | 0x01 0x00 | 1 Byte | 1 Byte |

This response is followed by a CMD_TXCOMPLETE_RSP to indicate the completion of the packet transmission. If the packet was correctly received by the remote module, it responds with a confirmation message containing the GPIO configurations. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_GET_CONFIG_RSP:

| Start signal | Command | Length | Remote address | RSSI | Status | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xEA | 2 Bytes | 4 Bytes | 1 Byte | 1 Byte | x Bytes | | x Bytes | 1 Byte |

CMD_GPIO_REMOTE_GET_CONFIG_RSP block structure

Each **Block** has the following format:

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

Length: Length of the subsequent bytes in this block

GPIO_ID: ID of the GPIO, see chapter 10.1

Function:

- 0x00:** GPIO disconnected
- 0x01:** GPIO works as input
- 0x02:** GPIO works as output

Values:

- if **Function** is disconnected, Length is 0x02:
value 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

7.6.4.1. Example: Read the current GPIO configuration of the remote device

Read the current GPIO configuration of the remote device with address 0x6C000002:

| Start signal | Command | Length | Remote address | CS |
|--------------|---------|-----------|---------------------|------|
| 0x02 | 0x2A | 0x04 0x00 | 0x02 0x00 0x00 0x6C | 0x2E |

The local module responds with a CMD_GPIO_REMOTE_GET_CONFIG_CNF followed by a CMD_TXCOMPLETE_RSP to indicate receipt of the packet over UART and transmission over the radio respectively. If the packet was correctly received by the remote module, it responds with a confirmation message containing the GPIO configurations. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_GET_CONFIG_RSP:

| Start signal | Command | Length | Remote address | RSSI | Blocks | CS |
|--------------|---------|--------|------------------------|------|--|------|
| 0x02 | 0xEA | 0x1A | 0x02 0x00 0x00 0x6C | 0xCD | 0x03 0x01 0x02 0x01 0x03 0x02 0x02 0x01 0x02 0x03 0x00 0x02 0x04 0x00 0x02 0x05 0x00 0x02 0x06 0x00 | 0x54 |

The GPIOs with GPIO_ID **0x01** and **0x02** are output high. The remaining GPIOs with GPIO_ID **0x03**, **0x04**, **0x05** and **0x06** are disconnected.

7.6.5. 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 correctly 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_WRITE_CNF):

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

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 10.1

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 10.1

Status:

- 0x00:** Success
- 0x01:** Failed

7.6.5.1. 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 | 0x27 | 0x03 0x00 | 0x02 0x01 0x00 | 0x25 |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | CS |
|--------------|----------------|-----------|--------|-----------------------|------|
| 0x02 | 0x67 | 0x04 0x00 | 0x00 | 0x02 0x01 0x00 | 0x62 |

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

7.6.6. CMD_GPIO_LOCAL_READ_REQ

This command reads the free GPIOs of the local device. This command can only be 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 | CS |
|--------------|---------|---------|--------------------|--------|
| 0x02 | 0x28 | 2 Bytes | x Bytes | 1 Byte |

Response (CMD_GPIO_LOCAL_READ_CNF):

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

CMD_GPIO_LOCAL_READ_REQ block structure

The **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 10.1

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 10.1

Value:

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

7.6.6.1. 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 | 0x28 | 0x03 0x00 | 0x02 0x01 0x02 | 0x27 |

Response:

| Start signal | Command 0x40 | Length | Status | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|--------|--------------------------|--------------------------|------|
| 0x02 | 0x68 | 0x07 0x00 | 0x00 | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0x6F |

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

7.6.7. CMD_GPIO_REMOTE_WRITE_REQ

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

Format:

| Start signal | Command | Length | Remote address | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------------------|-----|--------------------|--------|
| 0x02 | 0x2B | 2 Bytes | 4 Bytes | x Bytes | | x Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_WRITE_CNF):

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

This response is followed by a CMD_TXCOMPLETE_RSP to indicate the completion of the packet transmission. If the packet was correctly received by the remote module, it responds with a confirmation message. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_WRITE_RSP:

| Start signal | Command 0xC0 | Length | Remote address | RSSI | Block ₁ | ... | Block _n | CS |
|--------------|----------------|---------|----------------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xEB | 2 Bytes | 4 Byte | 1 Byte | x Bytes | | x Bytes | 1 Byte |

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 10.1

Value:

- if **Function** is output
 - 0x00:** Set GPIO to LOW
 - 0x01:** Set GPIO to HIGH

CMD_GPIO_REMOTE_WRITE_RSP 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 10.1

Status:

0x00: Success

0x01: Failed

7.6.7.1. Example: Set a remote output GPIO to LOW

Set the output GPIO (GPIO_ID **0x01**) of remote device with address 0x6C000002 to LOW:

| Start signal | Command | Length | Remote address | Block ₁ | CS |
|--------------|---------|-----------|------------------------|-----------------------|------|
| 0x02 | 0x2B | 0x07 0x00 | 0x02 0x00 0x00 0x6C | 0x02 0x01 0x00 | 0x43 |

The local module responds with a CMD_GPIO_REMOTE_WRITE_CNF followed by a CMD_TXCOMPLETE_RSP to indicate reception over UART and transmission over the radio. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_WRITE_RSP

| Start signal | Command | Length | Remote address | RSSI | Block ₁ | CS |
|--------------|---------|-----------|------------------------|------|-----------------------|------|
| 0x02 | 0xEB | 0x08 0x00 | 0x02 0x00 0x00 0x6C | 0xD0 | 0x02 0x01 0x00 | 0x5C |

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

7.6.8. CMD_GPIO_REMOTE_READ_REQ

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

Format:

| Start signal | Command | Length | Remote address | Block ₁ | CS |
|--------------|---------|---------|----------------|--------------------|--------|
| 0x02 | 0x2C | 2 Bytes | 4 Bytes | x Bytes | 1 Byte |

Response (CMD_GPIO_REMOTE_READ_CNF):

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

This response is followed by a CMD_TXCOMPLETE_RSP to indicate the completion of the packet transmission. If the packet was correctly received by the remote module, it responds with a confirmation message containing the GPIO values. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_READ_RSP:

| Start signal | Command | Length | Remote address | RSSI | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xEC | 2 Bytes | 4 Bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

CMD_GPIO_REMOTE_READ_REQ block structure

The **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 10.1

CMD_GPIO_REMOTE_READ_RSP 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 10.1

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.

7.6.8.1. Example: Read the values of remote GPIOs

Read the value of the GPIOs with GPIO_ID **0x01** and **0x02** of the remote device with address 0x6C000002:

| Start signal | Command | Length | Remote address | Block ₁ | CS |
|--------------|---------|-----------|---------------------|-----------------------|------|
| 0x02 | 0x2C | 0x07 0x00 | 0x02 0x00 0x00 0x6C | 0x02 0x01 0x02 | 0x46 |

The local module responds with a CMD_GPIO_REMOTE_READ_CNF followed by a CMD_TXCOMPLETE_RSP to indicate receipt of the packet over UART and completed transmission, respectively. If the packet was correctly received by the remote module, it responds with a confirmation message containing the GPIO values. The response from the remote module is forwarded to the host as CMD_GPIO_REMOTE_READ_RSP:

| Start signal | Command 0x40 | Length | Remote address | RSSI | Block ₁ | Block ₂ | CS |
|--------------|----------------|-----------|------------------------|------|-----------------------------|-----------------------------|------|
| 0x02 | 0xEC | 0x07 0x00 | 0x02 0x00 0x00 0x6C | 0xCE | 0x02 0x01 0x00 | 0x02 0x02 0x01 | 0x47 |

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

7.6.9. CMD_GPIO_REMOTE_SET_CONFIG_IND

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



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 | Remote address | RSSI | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xA9 | 2 Bytes | 4 bytes | 1 Byte | x Bytes | | x Bytes | 1 Byte |

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


7.6.9.1. Example: Two GPIOs have been configured by the remote device 0x6C00001 to output HIGH

| Start signal | Command | Length | Remote address | RSSI | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|------------------------|------|-------------------------------|-------------------------------|----|
| 0x02 | 0xA9 | 0x0D 0x00 | 0x01 0x00 0x00 0x6C | 0xCF | 0x03 0x01 0x02 0x01 | 0x03 0x02 0x02 0x01 | 07 |

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

7.6.10. CMD_GPIO_REMOTE_WRITE_IND

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




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 | Remote address | RSSI | Block ₁ | ... | Block _n | CS |
|--------------|---------|---------|----------------|--------|--------------------|-----|--------------------|--------|
| 0x02 | 0xAB | 2 Bytes | 4 Bytes | 1 Byte | 3 Bytes | | 3 Bytes | 1 Byte |

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



Note that the first byte of each block has a value of 0x03 although only two bytes follow.

7.6.10.1. Example: GPIOs have been written via remote access

| Start signal | Command | Length | Remote address | RSSI | Block ₁ | Block ₂ | CS |
|--------------|---------|-----------|------------------------|------|-----------------------------|-----------------------------|------|
| 0x02 | 0xAB | 0x0B 0x00 | 0x01 0x00 0x00 0x6C | 0xD1 | 0x03 0x01 0x00 | 0x03 0x02 0x01 | 0xAE |

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

7.7. Other messages

7.7.1. CMD_ERROR_IND

This indication is shown when the module entered an error state. In case of an error, all the bytes present in the buffer are discarded.

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 were aborted and UART has restarted. Restart the module, if the UART is still malfunctioning.

7.8. Run the radio test modes

The following commands describe configuration of the module in various test modes.



The commands for the test mode are solely intended for design verification and compliance tests. They should not be used under normal operation.

7.8.1. CMD_DTM_START_REQ

This command restarts the module in radio test mode. When starting in test mode, a `CMD_START_IND` 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.

As soon as the module is reset, the module leaves the test mode and normal operations can be performed.

Format:

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

Response (`CMD_DTM_START_CNF`):

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

7.8.2. CMD_DTM_REQ

This command starts and stops various test modes. To be able to run these test modes, the module has to be switched to the test mode using the `CMD_DTM_START_REQ`. After a test has been started, it has to be stopped first before a next test can be run.

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: 1 Mbit/s 2. 0x02: 2 Mbit/s 3. 0x03: Long range mode | 0x00 |

0x01: Start RX test

| Channel | Length | Payload |
|--|--------|---------|
| Frequency = (2403 + Channel * 2) MHz to be used for RX | 0x00 | 0x00 |

0x02: Start TX test

| Channel | Length | Payload |
|--|------------------------------|---|
| Frequency = (2403 + 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 = (2403 + Channel * 2) MHz to be used for TX | 0x00: Carrier test | 0x03: Vendor specific |
| TX power -40 up to +4 dBm (see chapter 9.10 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 successful

0x0001: Test failed

0x8000 + n: Received n packets during RX test

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

Start the transmission test on channel 0 (2403 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.

7.8.2.2. Example: Receiver, channel 0

Start the reception test on channel 0 (2403 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.

7.8.2.3. Example: Transmission, carrier test, channel 0

Start the carrier test on channel 0 (2403 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.

7.8.2.4. Example: Set TX power to -4 dBm

Set the TX power to -4 dBm (0xFC in two's complement notation (see chapter 9.10)):

| 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 |

7.8.2.5. Example: Set PHY to 2 Mbit/s mode

Set the phy to 2 Mbit/s 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 |

7.9. Messages Overview

Command bytes not shown in this table shall be considered RESERVED (internal use only).

| CMD | Message name | Description |
|------|--------------------------------|---|
| 0x00 | CMD_RESET_REQ | Reset the module |
| 0x01 | CMD_GETSTATE_REQ | Get the mode of operation |
| 0x02 | CMD_SLEEP_REQ | Go to sleep |
| 0x04 | CMD_UNICAST_DATA_REQ | Send unicast message to default destination address |
| 0x05 | CMD_MULTICAST_DATA_REQ | Send multicast message to default group ID |
| 0x06 | CMD_BROADCAST_DATA_REQ | Send broadcast message |
| 0x07 | CMD_UNICAST_DATA_EX_REQ | Send unicast message to the specified address |
| 0x08 | CMD_MULTICAST_DATA_EX_REQ | Send multicast message to the specified group ID |
| 0x09 | CMD_SETCHANNEL_REQ | Switch RF channel to the specified value (volatile) |
| 0x10 | CMD_GET_REQ | Get user setting |
| 0x11 | CMD_SET_REQ | Set user setting |
| 0x1B | CMD_TRANSPARENT_MODE_REQ | Set radio test |
| 0x1C | CMD_FACTORY_RESET_REQ | Restore factory settings |
| 0x1D | CMD_DTM_START_REQ | Retart in radio test mode |
| 0x1E | CMD_DTM_REQ | Set radio test |
| 0x25 | CMD_GPIO_LOCAL_SET_CONFIG_REQ | Configure local GPIO |
| 0x26 | CMD_GPIO_LOCAL_GET_CONFIG_REQ | Get local GPIO configuration |
| 0x27 | CMD_GPIO_LOCAL_WRITE_REQ | Write value of a local GPIO pin |
| 0x28 | CMD_GPIO_LOCAL_READ_REQ | Read value of a local GPIO pin |
| 0x29 | CMD_GPIO_REMOTE_SET_CONFIG_REQ | Configure remote GPIO |
| 0x2A | CMD_GPIO_REMOTE_GET_CONFIG_REQ | Get remote GPIO configuration |
| 0x2B | CMD_GPIO_REMOTE_WRITE_REQ | Write value of a remote GPIO pin |
| 0x2C | CMD_GPIO_REMOTE_READ_REQ | Read value of a remote GPIO pin |

Table 17: Requests

| CMD | Message name | Description |
|------|------------------|------------------------|
| 0x40 | CMD_RESET_CNF | Reset request received |
| 0x41 | CMD_GETSTATE_CNF | Mode requested |

| | | |
|------|--------------------------------|--------------------------------------|
| 0x42 | CMD_SLEEP_CNF | Sleep request received |
| 0x44 | CMD_DATA_CNF | Send data received |
| 0x49 | CMD_SETCANNEL_CNF | Set channel received |
| 0x50 | CMD_GET_CNF | User setting get received |
| 0x51 | CMD_SET_CNF | User setting set received |
| 0x5B | CMD_TRANSPARENT_MODE_CNF | Restart in transparent mode received |
| 0x5C | CMD_FACTORY_RESET_CNF | Factory reset received |
| 0x5D | CMD_DTM_START_CNF | Restart in radio test mode received |
| 0x5E | CMD_DTM_CNF | Set radio test received |
| 0x65 | CMD_GPIO_LOCAL_SET_CONFIG_CNF | Local GPIO config set received |
| 0x66 | CMD_GPIO_LOCAL_GET_CONFIG_CNF | Local GPIO config get received |
| 0x67 | CMD_GPIO_LOCAL_WRITE_CNF | Local GPIO write received |
| 0x68 | CMD_GPIO_LOCAL_READ_CNF | Local GPIO read received |
| 0x69 | CMD_GPIO_REMOTE_SET_CONFIG_CNF | Remote GPIO config set received |
| 0x6A | CMD_GPIO_REMOTE_GET_CONFIG_CNF | Remote GPIO config get received |
| 0x6B | CMD_GPIO_REMOTE_WRITE_CNF | Remote GPIO write received |
| 0x6C | CMD_GPIO_REMOTE_READ_CNF | Remote GPIO read received |

Table 18: Confirmations

| CMD | Message name | Description |
|------|--------------------------------|---|
| 0x73 | CMD_START_IND | Module ready to receive commands |
| 0x84 | CMD_DATA_IND | Data received over radio |
| 0x99 | CMD_SNIFFER_IND | Received radio messages in sniffer mode |
| 0xA2 | CMD_ERROR_IND | Error indication message |
| 0xA9 | CMD_GPIO_REMOTE_SET_CONFIG_IND | Local GPIO config modified by remote host |
| 0xAB | CMD_GPIO_REMOTE_WRITE_IND | Local GPIO state by remote host |
| 0xC4 | CMD_TXCOMPLETE_RSP | Data Transmit complete |
| 0xE9 | CMD_GPIO_REMOTE_SET_CONFIG_RSP | Remote module responded for GPIO set config request |
| 0xEA | CMD_GPIO_REMOTE_SET_CONFIG_RSP | Remote module responded for GPIO get config request |
| 0xEB | CMD_GPIO_REMOTE_WRITE_RSP | Remote module responded for GPIO write request |

| | | |
|------|--------------------------|---|
| 0xEC | CMD_GPIO_REMOTE_READ_RSP | Remote module responded for GPIO read request |
|------|--------------------------|---|

Table 19: Indications

8. The transparent interface

In this mode, the Thyone-e acts as a radio bridge. Any data received on the UART will be sent via radio. The receiver may operate in command mode or in transparent mode. The main application for the transparent mode is a serial cable replacement.



For new system designs, we highly recommend using the command mode.

8.1. Entering the transparent mode

The transparent mode can be entered using one of the following methods:

- when `UART_MODE` is configured to `0x00` and `MODE_1` pin is HIGH during boot-up.
- when `UART_MODE` is configured to `0x01` using the command `CMD_TRANSPARENT_MODE_REQ`.
- when `UART_MODE` is configured to `0x02`.

8.2. Leaving the transparent mode

The transparent mode can be left using one of the following methods:

- when `UART_MODE` is configured to `0x00`: performing a module reset and holding `MODE_1` pin to LOW during boot-up.
- when `UART_MODE` is configured to `0x01`: performing a module reset.
- when `UART_MODE` is configured to `0x02` and `UART_TRANSP_ESC_ENABLE` is enabled (default), by using the escape sequence `UART_TRANSP_ESC` (default: "+++").

8.2.1. Transparent mode escape sequence

The transparent mode escape sequence requires a pause of > 50 ms before and after the configured 3 byte sequence (default "+++") to be accepted.

A pause in between two consecutive bytes of the escape sequence shall not be larger than a byte duration or 5 ms - whichever is smaller.



Figure 11: Transparent mode escape sequence timing

The corresponding parameters are `UART_TRANSP_ESC_ENABLE` for enabling and disabling this sequence and `UART_TRANSP_ESC` for the byte sequence itself.



When `UART_MODE` is configured to `0x02`, make sure that the parameter `UART_TRANSP_ESC_ENABLE` is enabled and that a triplet of escape characters is configured. Otherwise, there is no possibility to leave the transparent mode.

8.3. Transparent mode configuration

In addition to the escape sequence, the following parameters can be configured to control the operation of the module in transparent mode. See chapter 9 for details.

- **Flow control:** The Thyone-e implements additional *BUSY* pin to indicate that the module is busy doing radio tasks and hence cannot accept any more data on the UART. It is mandatory to implement the Thyone-e *BUSY* pin and enable the UART flow control on the host and Thyone-e, in order to prevent data loss on the UART (Figure 12). In the absence of this flow control scheme, the application must be able to accept/handle loss of data.

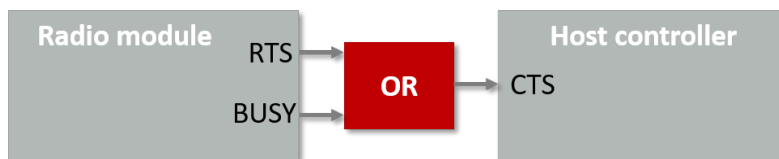


Figure 12: Transparent mode flow control

- **Timeout:** The parameter determines the time until which the module has to wait on data from the UART, before triggering a data transmission.(see `UART_TRANSPARENT_TIMEOUT`).
- **End of transmission characters:** The radio transmission of the data in the UART buffer can also be triggered on reception of a pre-configured set of ETX characters. The parameters `UART_TRANSP_ETX_CONFIG` and `UART_TRANSP_ETX` have to be configured to use this feature.
- **Addressing:** The network addressing is also available in transparent mode. The module can be pre-configured to broadcast, multicast to a pre-defined Group-ID or unicast to a specific address, by configuring the parameter `MAC_TRANSPARENT_ADDR_MODE`

8.4. Restrictions in transparent mode

The transparent mode can only provide a basic set of functionalities due to the lack of control commands.

- No access to user settings and runtime settings: The module configuration, such as UART settings (baud rate, hardware flow control), radio profile and radio channel, addressing and encrypting information, are taken from the non-volatile user settings. The settings can only be modified in command mode.
- A start in transparent mode does not maintain previously configured volatile values ("Runtime settings"). Any volatile configuration is lost when toggling to or starting in transparent mode.
- No status information via the communication interface. The user does not get any status, error or confirmation messages from the Thyone-e.
- Data streaming: when the host does not implement hardware flow control and adheres to the Thyone-e BUSY pin, data on the UART towards the module may be lost.

9. 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.



When the configuration settings of the module are known in advance, an individualized firmware can be used to prevent in-field configuration. See chapter 13 for more information regarding firmware individualization.

9.1. SERIAL_NUMBER: Read the serial number of the module

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------|--------------------|---------------|-------------|-----------------|
| 1 | SERIAL_NUMBER | - | - | read | 4 |

This setting contains the serial number of the module.

9.1.1. Example 1

Request the serial number of the module using `CMD_GET_REQ` with settings index 01:

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

Response `CMD_GET_CNF`: Successfully read out the serial number, it is 0x6C000001:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---------------------|------|
| 0x02 | 0x50 | 0x05 0x00 | 0x00 | 0x01 0x00 0x00 0x6C | 0x3A |

9.2. FW_Version: Read the firmware version

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 2 | FW_VERSION | - | - | read | 3 |

This setting contains the firmware version of the module.

9.2.1. 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 | 0x02 | 0x11 |

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 LSB first):

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

9.3. UART_CONFIG: Modify the UART speed

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 4 | UART_CONFIG | 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_CONFIG | 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 |
| 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 |



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.



As can be seen in table above, a higher baud rate than 230400 baud is not available without the usage of UART flow control. This is to avoid data loss with fast baud rates due to high HF-activity of the chip. Should the end user experience loss of bytes with a lower baud rate than 230400, activating UART flow control might be an option.

9.3.1. Example 1

Set the baud rate index to 0x0F (38400 Baud with flow control and parity none) using CMD_SET_REQ with settings index 4:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x04 | 0x0F | 0x1A |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.3.2. Example 2

Request the baud rate index 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 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 |

9.4. UART_MODE: Mode of UART operation

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 5 | UART_MODE | See description | 0 | read/write | 1 |

This parameter defines the mode of operation of the module's UART.

| Value | Mode |
|-------|---|
| 0 | Pin <i>MODE_1</i> determines the mode of operation. (0 = command mode, 1 = Transparent mode). |
| 1 | Command mode (see chapter 7). |
| 2 | Transparent mode (see chapter 8). |

9.4.1. Example 1

Set the UART mode to pin based switch:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x05 | 0x00 | 0x14 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.4.2. Example 2

Request the current configuration of the UART mode:

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

Response CMD_GET_CNF: Successfully read out the value 0x00, UART mode determined by the level of *MODE_1* on boot-up:

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

9.5. UART_TRANSPARENT_TIMEOUT: Mode of UART operation

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------------------|--------------------|---------------|-------------|-----------------|
| 6 | UART_TRANSPARENT_TIMEOUT | 2-255 | 10 | read/write | 1 |

In the transparent mode, this parameter defines the duration (in ms) after the last received byte over UART, for which the module waits before triggering a transmit of the data in the buffer over the radio.

9.5.1. Example 1

Set the UART transparent timeout to 10 ms:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x06 | 0x0A | 0x1D |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.5.2. Example 2

Request the current configuration of the UART mode:

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

Response CMD_GET_CNF: Successfully read out the value 10 ms:

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

9.6. RF_CHANNEL: Channel (Frequency) of operation

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 7 | RF_CHANNEL | 0-39 | 21 | read/write | 1 |

This parameter defines the radio channel and hence the frequency of operation of the radio module (Table 21).

| Index | Operating frequency [MHz] | Index | Operating frequency [MHz] |
|-------|---------------------------|-------|---------------------------|
| 0 | 2402 | 20 | 2442 |
| 1 | 2403 | 21 | 2444 |
| 2 | 2406 | 22 | 2446 |
| 3 | 2408 | 23 | 2448 |
| 4 | 2410 | 24 | 2450 |
| 5 | 2412 | 25 | 2452 |
| 6 | 2414 | 26 | 2454 |
| 7 | 2416 | 27 | 2456 |
| 8 | 2418 | 28 | 2458 |
| 9 | 2420 | 29 | 2460 |
| 10 | 2422 | 30 | 2462 |
| 11 | 2424 | 31 | 2464 |
| 12 | 2426 | 32 | 2466 |
| 13 | 2428 | 33 | 2468 |
| 14 | 2430 | 34 | 2470 |
| 15 | 2432 | 35 | 2472 |
| 16 | 2434 | 36 | 2474 |
| 17 | 2436 | 37 | 2476 |
| 18 | 2438 | 38 | 2478 |
| 19 | 2440 | 39 | 2480 |

Table 21: Channel index and Frequency of operation

9.6.1. Example 1

Set the RF channel to channel 10:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x07 | 0x0A | 0x1C |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.6.2. Example 2

Request the current configuration of RF channel:

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

Response CMD_GET_CNF: Successfully read out the default RF channel index, 21:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x15 | 0x45 |

9.7. ENCRYPTION_MODE: Set encryption mode

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------|--------------------|---------------|-------------|-----------------|
| 8 | ENCRYPTION_MODE | 0-3 | 0 | read/write | 1 |

This parameter determines the level of encryption on the module. The table below contains possible values as well as a description of the transmission and reception behavior in each mode.

| ENCRYPTION_MODE | Transmission | Reception |
|-----------------|--------------|---|
| 0x00 | Unencrypted | Unencrypted and encrypted packets (with matching key). |
| 0x01 | Encrypted | Unencrypted and encrypted packets (with matching key). |
| 0x02 | Unencrypted | Encrypted packets only (unencrypted and non-decryptable packets are discarded). |
| 0x03 | Encrypted | Encrypted packets only (unencrypted and non-decryptable packets are discarded). |

9.7.1. Example 1

Set the encryption mode to encrypt all transmit messages:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x08 | 0x01 | 0x18 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.7.2. Example 2

Request the current configuration of RF encryption mode:

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

Response CMD_GET_CNF: Successfully read out the encryption mode, 00:

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

9.8. RF_PROFILE: Modify the radio profile

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 9 | RF_PROFILE | 2-3 | 2 | read/write | 1 |

This setting determines the modulation and coding used on the physical layer of the module. The available radio profile are listed in the following table.

| Radio profile | Description |
|---------------|-------------------|
| 2 | 1000 kbit/s mode. |
| 3 | 2000 kbit/s mode. |

9.8.1. Example 1

Set the module in high throughput 2 Mbps mode:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x09 | 0x03 | 0x1B |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.8.2. Example 2

Request the current radio profile of the module:

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

Response CMD_GET_CNF: Successfully read out the value 2 = 1 Mbps long range mode:

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

9.9. RF_NUM_RETRIES: Number of retries

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|----------------|--------------------|---------------|-------------|-----------------|
| 10 | RF_NUM_RETRIES | 0-255 | 3 | read/write | 1 |

This value determines the number of times the module tries to transmit an RF packet before dropping it. A value greater than 0 automatically prompts the receiver to send an "ACK". The transmitter retries transmission up to a maximum of RF_NUM_RETRIES times until it receives an ACK.



Choosing a value greater than 0 prompts an ACK from the receiver, which will increase the overall traffic on the radio channel. The number of retries has to be chosen prudently to avoid blocking the channel.

9.9.1. Example 1

Set the number of retries to 1:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x0A | 0x01 | 0x1A |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.9.2. Example 2

Request the current configuration of the number of retries:

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

Response CMD_GET_CNF: Successfully read out the value = 3:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x03 | 0x53 |

9.10. RF_TX_POWER: Modify the output power

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bsytes |
|----------------|-------------|--------------------|---------------|-------------|------------------|
| 11 | RF_TX_POWER | 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 | -4 | 0 | 4 |
| Two's complement, hexadecimal | 0xD8 | 0xEC | 0xF0 | 0xF4 | 0xF8 | 0xFC | 0x00 | 0x04 |

9.10.1. 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 11:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x0B | 0xF8 | 0xE2 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.10.2. Example 2

Request the output power 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 0x04 = 4 dBm:

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

9.11. RF_REPEATER_THRESHOLD: Modify the repeater threshold

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------------|--------------------|---------------|-------------|-----------------|
| 29 | RF_REPEATER_THRESHOLD | -100 - 8 | 8 | read/write | 1 |

This setting determines the threshold used, when the repeater mode is enabled. The value has to be entered in hexadecimal and as two's complement. In case the repeater is enabled, the radio frame is only relayed if its RSSI value is below this threshold. Otherwise, the message is discarded.

9.11.1. Example 1

Set the threshold of the module to -50 dBm, which is 0xCE in two's complement notation, using CMD_SET_REQ with settings index 29:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x1D | 0xCE | 0xC2 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.11.2. Example 2

Request the threshold of the module using CMD_GET_REQ with settings index 29:

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


Response CMD_GET_CNF: Successfully read out the value 0x08 = 8 dBm:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x08 | 0x58 |

9.12. RF_RP_NUM_SLOTS: Number of retries

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------|--------------------|---------------|-------------|-----------------|
| 12 | RF_RP_NUM_SLOTS | 1-255 | 32 | read/write | 1 |

This value contains the number of time slots to be used for the packet repetition. When using several repeater devices in a single network, repeated data packets may collide on the radio channel, when all repeater devices send the received packet at the same time. To avoid this, the frequency channel is divided in RF_RP_NUM_SLOTS time slots, where each repeater chooses a certain slot randomly.



This calculation assumes exactly one radio frame over the whole duration of the repeating, i.e. a low throughput scenario. With higher throughput requirements, the Poisson distribution of frames together with the selected channel access scheme, needs to be included in the calculation.

The smallest number of time slots that is needed, depends on the network structure and the number of repeaters used. Assuming there are NumRP repeater devices in the range of a sending device and there is only one radio packet present over the entire duration of all repeating, the probability of at least two of the same repeated packet collides can be calculated by:

$$1 - \frac{RP_NumSlots!}{RP_NumSlots^{NumRP} \times (RP_NumSlots - NumRP)!}$$

Common values are:

| NumRP | RF_RP_NUM_SLOTS | Collision probability |
|-------|-----------------|-----------------------|
| 2 | 32 | 3.1% |
| 3 | 32 | 9.2% |
| 4 | 32 | 17.7% |
| 5 | 64 | 14.8% |
| 6 | 64 | 21.5% |
| 7 | 128 | 15.4% |

In the example network shown in figure 22, there are only two repeaters that can conflict with each other. Repeater 2 and 3 are forwarding the packet received from Sender 1 "at the same time". Thus, NumRP equal 2 and RF_RP_NUM_SLOTS equal 32 is sufficient to have a collision probability of less than 5 %. The time delay used by the repeater device can be determined as the time needed to send one packet (see table 13) times a random number between one and RF_RP_NUM_SLOTS.

Example:

In RF_PROFILE 2, the maximum send time for one packet is about 3 ms. If we use 32 RF_RP_NUM_SLOTS, the packet is forwarded at the latest after 96 ms (32 × 3 ms).

9.12.1. Example 1

Set the number of repeater slots to 32:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x0C | 0x20 | 0x3D |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.12.2. Example 2

Request the current configuration of the number of repeater slots:

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

Response CMD_GET_CNF: Successfully read out the value = 32:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x20 | 0x70 |

9.13. MAC_SOURCE_ADDRESS: Modify the source address of the module

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------------|--------------------|------------------------|-------------|-----------------|
| 16 | MAC_SOURCE_ADDRESS | - | 0xFF 0xFF 0xFF 0xFF | read/write | 4 |

This setting contains the source address of the radio module. The address is included in the radio packets transmitted by the module and has to be unique in a network of devices. If not configured (i.e. if set to the default value), the module uses its SERIAL_NUMBER as its source address. The address has to be specified in LSB first format.

9.13.1. Example 1

Set the source address of the module to 0x6C000002:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|---------------------|------|
| 0x02 | 0x11 | 0x05 0x00 | 0x10 | 0x02 0x00 0x00 0x6C | 0x68 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.13.2. Example 2

Request the source address 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 source address, it is 0x6C000002:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|---------------------|------|
| 0x02 | 0x50 | 0x05 0x00 | 0x00 | 0x02 0x00 0x00 0x6C | 0x39 |

9.14. MAC_DEST_ADDRESS: Modify the default destination address

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------|--------------------|------------------------|-------------|-----------------|
| 17 | MAC_DEST_ADDRESS | - | 0xFF 0xFF 0xFF 0xFF | read/write | 4 |

This setting contains the default destination address of the radio module. The address is used by default as the destination address, when sending data using the CMD_UNICAST_DATA_REQ. The address has to be specified in LSB first format.

9.14.1. Example 1

Set the destination address of the module to 0x6C000002:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|---------------------|------|
| 0x02 | 0x11 | 0x05 0x00 | 0x11 | 0x02 0x00 0x00 0x6C | 0x69 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.14.2. Example 2

Request the destination address 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 default destination address, it is 0x6C000002:

| Start signal | Command | Length | Status | Parameter | CS |
|--------------|---------|-----------|--------|---------------------|------|
| 0x02 | 0x50 | 0x05 0x00 | 0x00 | 0x02 0x00 0x00 0x6C | 0x39 |

9.15. MAC_GROUP_ID: Modify the default group ID

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------|--------------------|---------------|-------------|-----------------|
| 18 | MAC_GROUP_ID | - | 0x00 | read/write | 1 |

This setting contains the default destination group ID of the radio module. This group ID is used by default as the multicast address, when sending data using the `CMD_MULTICAST_DATA_REQ`.

9.15.1. Example 1

Set the group ID of the module to 0xAA:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x12 0xAA | 0xA9 |

Response `CMD_SET_CNF`: Successfully modified the setting:

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

9.15.2. Example 2

Request the multicast group ID 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 multicast group ID, it is 0xAA:

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

9.16. MAC_TRANSPARENT_ADDR_MODE: Modify the addressing mode used in transparent mode

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|---------------------------|--------------------|---------------|-------------|-----------------|
| 19 | MAC_TRANSPARENT_ADDR_MODE | 0-2 | 0 | read/write | 1 |

This setting determines the addressing mode used in the transparent mode. The following table describes the permissible values:

| value | Description |
|-------|------------------------------|
| 0 | Broadcast. |
| 1 | Multicast to MAC_GROUP_ID. |
| 2 | Unicast to MAC_DEST_ADDRESS. |

9.16.1. Example 1

Set the transparent mode addressing to broadcast:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x13 | 0x00 | 0x02 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.16.2. Example 2

Request the current value of the setting:

| Start signal | Command | Length | Settings index | CS |
|--------------|---------|-----------|----------------|------|
| 0x02 | 0x10 | 0x01 0x00 | 0x13 | 0x00 |

Response CMD_GET_CNF: Successfully read out the value 0 = broadcast addressing:

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

9.17. MAC_ENCRYPTION_KEY: Set the key used for encryption/decryption

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|--------------------|--------------------|---------------|-------------|-----------------|
| 20 | MAC_ENCRYPTION_KEY | - | 0 | write | 16 |

This setting contains the key used for encryption/decryption of the user payload. This is a write-only parameter and cannot be read-out over the UART.

A key consisting of 16 times 0x00 is considered as "no key set". Thus, write 16 times 0x00 to delete your key.



Security warning: The key is stored on the flash of the module and can be potentially compromised when physical access to the module is possible.

9.17.1. Example 1

Set the encryption key to 0x11223344556677889900112233445566:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|--|------|
| 0x02 | 0x11 | 0x11 0x00 | 0x14 | 0x1122334455667788 9900112233445566 | 0x70 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.18. MAC_TTL: Time to live

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 21 | MAC_TTL | 0-255 | 3 | read/write | 1 |

This defines the time-to-live value of the packets transmitted over the radio. The time-to-live packet defines the maximum number of hops that a packet can make before being discarded. A limited number of hops prevents indefinite forwarding of a packet over a network with multiple repeaters.

9.18.1. Example 1

Set the time-to-live value to 3:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x15 | 0x03 | 0x07 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.18.2. Example 2

Request the current configuration of time-to-live:

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

Response CMD_GET_CNF: Successfully read out the value = 3:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x03 | 0x53 |

9.19. GPIO_BLOCK_REMOTE_GPIO_CONFIG

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------------------------|--------------------|---------------|-------------|-----------------|
| 24 | GPIO_BLOCK_REMOTE_GPIO_CONFIG | 0/1 | 0 | read/write | 1 |

The Thyone-e module allows configuration and control of some of its digital I/O pins remotely. This parameter, when enabled, blocks remote configuration of these digital I/O pins.

| Value | Mode |
|-------|---|
| 0 | Allow remote configuration (default value). |
| 1 | Disable remote configuration. |

9.19.1. Example 1

Disable remote I/O configuration on the module:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x18 | 0x01 | 0x08 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.19.2. Example 2

Request the current value of the parameter at index 24:

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

Response CMD_GET_CNF: Successfully read out the value = 1:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x01 | 0x51 |

9.20. UART_TRANSP_ETX_CONFIG: Trigger radio transmit on ETX characters

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------------|--------------------|---------------|-------------|-----------------|
| 25 | UART_TRANSP_ETX_CONFIG | See description | 0 | read/write | 1 |

The Thyone-e supports a transparent mode where the module acts as transparent radio bridge. This parameter enables triggering of a radio transmission in the transparent mode when a (set of) pre-configured ETX (end of transmission) character(s) are received over the UART. The parameter UART_TRANSP_ETX contains the characters to trigger transmission. Refer to chapter 8 for further details.

| Bit no. | Description |
|---------|---|
| 1 : 0 | Configuration whether 1 byte or 2 byte ETX is used. |
| | 0x0 No trigger ETX trigger is disabled. |
| | 0x1 1 byte ETX. Received UART data is handled when ETX matches first byte of UART_TRANSP_ETX. |
| | 0x2 2 byte ETX. Received UART data is handled when ETX matches both bytes of UART_TRANSP_ETX. |
| 2 | TRANSP_REMOVE_ETX: If this bit is set, the ETX (1 byte or 2 bytes) is removed from the payload data, before relaying the data to the radio. |
| 7 : 3 | Reserved. |

Table 22: ETX configuration flags

9.20.1. Example 1

Set the ETX mode to 2 byte:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x19 | 0x02 | 0x08 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.20.2. Example 2

Request the current value of the parameter at index 25.

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

Response CMD_GET_CNF: Successfully read out the value = 2.

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

9.21. UART_TRANSP_ETX: ETX characters

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------|--------------------|---------------|-------------|-----------------|
| 26 | UART_TRANSP_ETX | - | 0x0A 0x0D | read/write | 2 |

This parameter defines the characters on which transmission can be triggered in transparent mode. Based on the value of the parameter UART_TRANSP_ETX_CONFIG, 1 or 2 byte ETX can be used. Refer to chapter 8 for further details.

9.21.1. Example 1

Set the ETX characters to 0x0A0D:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x03 0x00 | 0x1A | 0x0A 0x0D | 0x0D |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.21.2. Example 2

Request the current value of the parameter at index 26:

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

Response CMD_GET_CNF: Successfully read out the value = 0x0A0D:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x03 0x00 | 0x00 | 0x0A 0x0D | 0x52 |

9.22. UART_TRANSP_ESC_ENABLE: Enable switch to command mode on escape sequence

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|------------------------|--------------------|---------------|-------------|-----------------|
| 27 | UART_TRANSP_ESC_ENABLE | 0/1 | 1 | read/write | 1 |

This parameter enables/disables switching from transparent mode to command mode on receiving an escape sequence over the UART. Refer to chapter 8 for further details.

| Value | Mode |
|-------|---------------------------------------|
| 0 | Disable switch on ESC. |
| 1 | Enable switch on ESC (default value). |

9.22.1. Example 1

Enable escape from transparent mode:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x1B | 0x01 | 0x0B |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.22.2. Example 2

Request the current value of the parameter at index 27:

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

Response CMD_GET_CNF: Successfully read out the value = 1:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|-----------|------|
| 0x02 | 0x50 | 0x02 0x00 | 0x00 | 0x01 | 0x51 |

9.23. UART_TRANSP_ESC: Escape sequence

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-----------------|--------------------|----------------------|-------------|-----------------|
| 28 | UART_TRANSP_ESC | - | 0x2B 0x2B 0x2B | read/write | 3 |

This parameter defines the set of three characters on which the module is switched from transparent mode to command mode. The parameter UART_TRANSP_ESC_ENABLE has to be set to activate this feature. Refer to chapter 8 for further details.

9.23.1. Example 1

Set the ETX characters to 0x2B2B2B ("+++"):

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|----------------|------|
| 0x02 | 0x11 | 0x04 0x00 | 0x1C | 0x2B 0x2B 0x2B | 0x20 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.23.2. Example 2

Request the current value of the parameter at index 28:

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

Response CMD_GET_CNF: Successfully read out the value = 0x0A0D:

| Start signal | Command 0x40 | Length | Status | Parameter | CS |
|--------------|----------------|-----------|--------|----------------|------|
| 0x02 | 0x50 | 0x04 0x00 | 0x00 | 0x2B 0x2B 0x2B | 0x52 |

9.24. MODULE_MODE: Mode of radio operation

| Settings index | Designation | Permissible values | Default value | Permissions | Number of bytes |
|----------------|-------------|--------------------|---------------|-------------|-----------------|
| 32 | MODULE_MODE | 0,1,2 | 0 | read/write | 1 |

This parameter defines the mode of operation of the module's radio. The table below contains the permissible values and descriptions for the same:

| Value | Mode |
|-------|------------------------|
| 0 | Normal mode (Default). |
| 1 | Sniffer mode. |
| 2 | Repeater mode. |

9.24.1. Example 1

Configure the module in repeater mode:

| Start signal | Command | Length | Settings index | Parameter | CS |
|--------------|---------|-----------|----------------|-----------|------|
| 0x02 | 0x11 | 0x02 0x00 | 0x20 | 0x02 | 0x33 |

Response CMD_SET_CNF: Successfully modified the setting:

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

9.24.2. Example 2

Request the current value of the parameter at index 32:

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

Response CMD_GET_CNF: Successfully read out the value = 2:

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

| Index | Designation | Summary | Permissible values | Default value | Permission | Size (bytes) |
|-------|-------------------------------|------------------------------|--------------------|------------------------------|--------------|--------------|
| 1 | SERIAL_NUMBER | Serial number | - | - | read | 4 |
| 2 | FW_VERSION | Firmware version | - | - | read | 4 |
| 4 | UART_CONFIG | Baud rate of the UART | See description | 22 | write / read | 1 |
| 5 | UART_MODE | UART mode of the module | 0-2 | 0 | write / read | 1 |
| 6 | UART_TRANSPARENT_TIMEOUT | Time out in transparent mode | 2 - 255 | 10 | read / write | 1 |
| 7 | RF_CHANNEL | RF channel | 0 - 39 | 21 | read / write | 1 |
| 8 | ENCRYPTION_MODE | Set encryption mode | 0 - 3 | 0 | read / write | 1 |
| 9 | RF_PROFILE | RF profiles | 2-3 | 2 | read / write | 1 |
| 10 | RF_NUM_RETRIES | Number of retries | 0-255 | 3 | read / write | 1 |
| 11 | RF_TX_POWER | Transmit power | See description | 4 | read / write | 1 |
| 12 | RF_RP_NUM_SLOTS | | See description | 32 | read / write | 1 |
| 16 | MAC_SOURCE_ADDRESS | Source address | - | 0xFF 0xFF 0xFF 0xFF | read / write | 4 |
| 17 | MAC_DEST_ADDRESS | Destination address | - | 0xFF 0xFF 0xFF 0xFF | read / write | 4 |
| 18 | MAC_GROUP_ID | Group ID | - | 0x00 | read / write | 1 |
| 19 | MAC_TRANSPARENT_ADDR_MODE | | 0-2 | 0 | read / write | 1 |
| 20 | MAC_ENCRYPTION_KEY | 16 byte key | See description | 0 | write | 16 |
| 21 | MAC_TTL | Time to live | 0 - 255 | 3 | read / write | 1 |
| 24 | GPIO_BLOCK_REMOTE_GPIO_CONFIG | Block GPIO remote access | 0/1 | 0 | read / write | 1 |

| | | | | | | |
|----|------------------------|----------------------------|-----------------|----------------------|--------------|---|
| 25 | UART_TRANSP_ETX_CONFIG | Enable/disable ETX | See description | 0 | read / write | 1 |
| 26 | UART_TRANSP_ETX | ETX characters | See description | 0A 0D | read / write | 2 |
| 27 | UART_TRANSP_ESC_ENABLE | Escape to CMD | 0/1 | 0 | read / write | 1 |
| 28 | UART_TRANSP_ESC | Escape character | See description | 0x2B 0x2B 0x2B | read / write | 3 |
| 29 | RF_REPEATER_THRESHOLD | Threshold for packet relay | -100 - 8 | 8 | read / write | 1 |
| 32 | MODULE_MODE | mode | 0,1,2 | 0 | read / write | 1 |

Table 23: Table of settings

10. Remote GPIO control

The Thyone-e allows to configure and control special GPIOs via remote and local access. Chapter 7.6 contains the description of the necessary commands.

To use the remote GPIO control feature of the Thyone-e, the GPIOs of interest must be configured first. This can be done in two ways. Either by the local host (see figure 13), or via a remote device (see figure 14).

In case of the local host, it must send a `CMD_GPIO_LOCAL_SET_CONFIG_REQ` command to the radio module via UART. In case of the remote device, it must send a unicast message `CMD_GPIO_REMOTE_SET_CONFIG_REQ` command to the radio module that shall be configured.

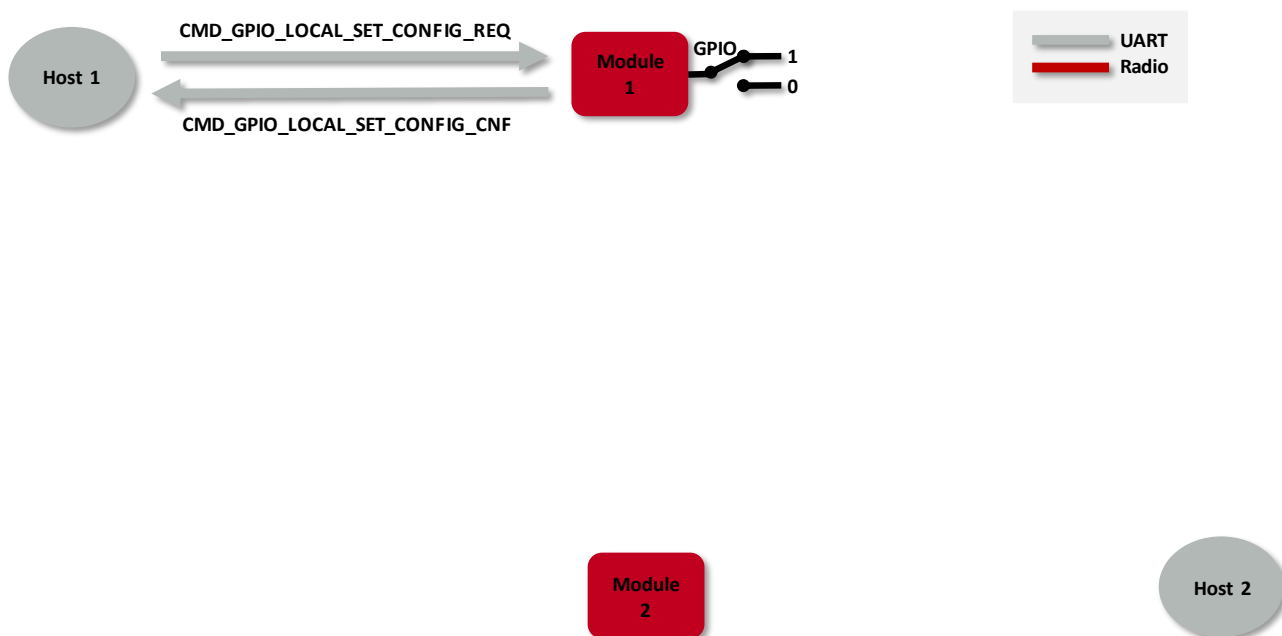


Figure 13: Configure the local GPIOs via local host

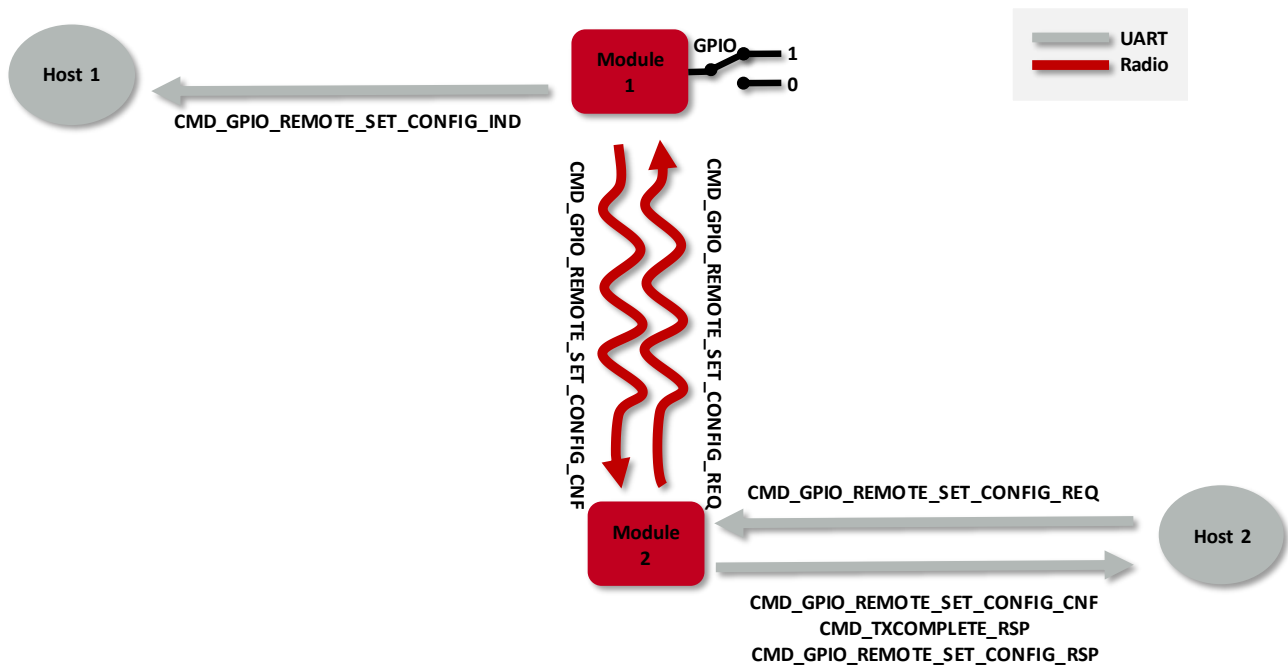


Figure 14: Configure the local GPIOs via remote device host

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_FACTORY_RESET_REQ` command. The currently active configuration can be also requested using the respective commands, `CMD_GPIO_LOCAL_GET_CONFIG_REQ` via local host or `CMD_GPIO_REMOTE_GET_CONFIG_REQ` via remote device (see figures 15 and 16).

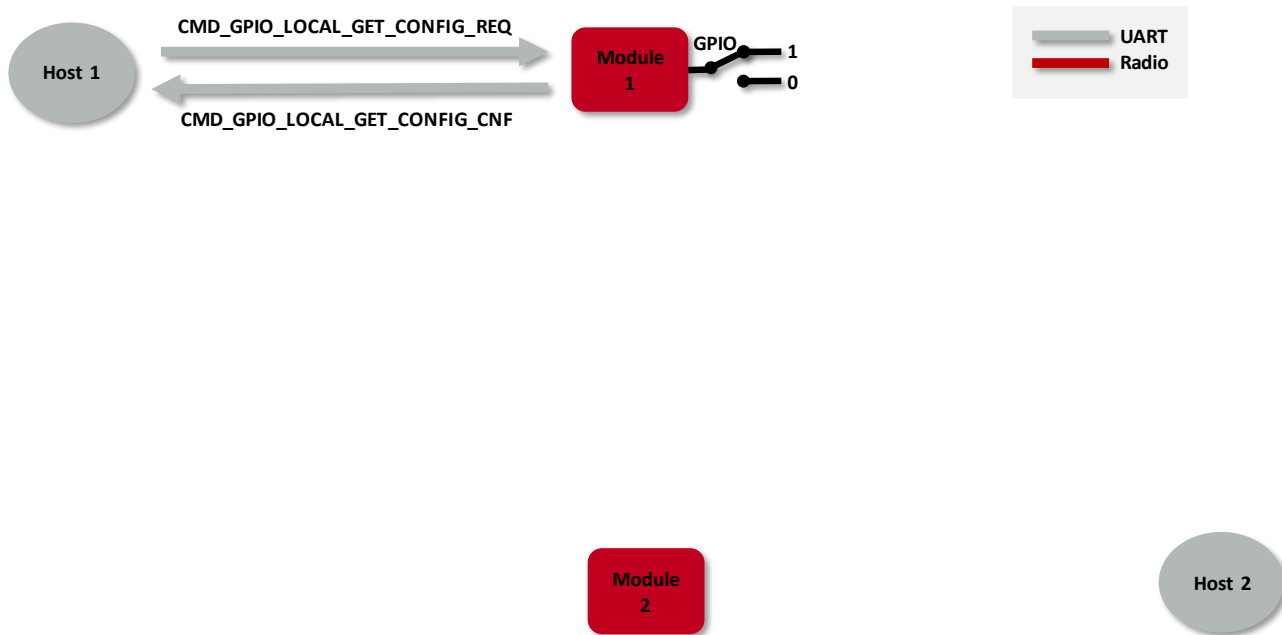


Figure 15: Read the configuration of the local GPIOs via local host

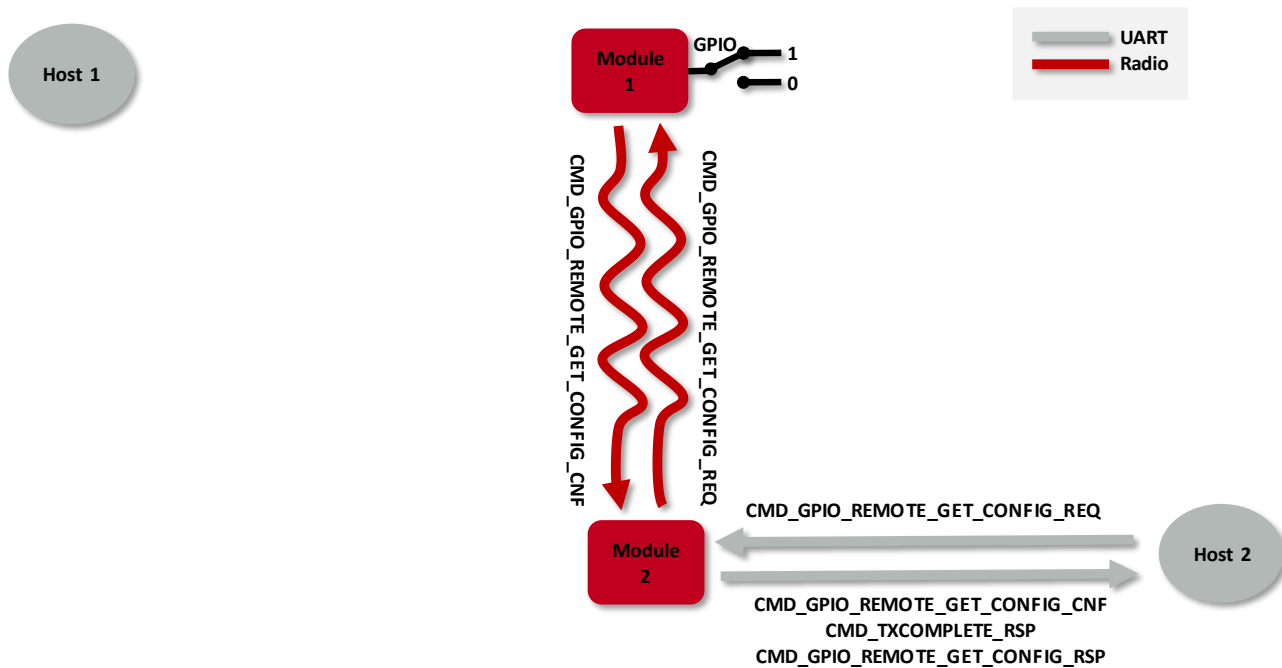


Figure 16: Read the configuration of the local GPIOs via remote device host



The SET_CONFIG command will consume a flash cycle in the device that is subject to the change. It is recommended to perform a CMD_GPIO_LOCAL_GET_CONFIG_REQ or CMD_GPIO_REMOTE_GET_CONFIG_REQ first and check if a change is required before performing the SET_CONFIG command.

For security reasons the remote configuration and remote access can be locked by using the GPIO_BLOCK_REMOTE_GPIO_CONFIG parameter in the user settings (default: allow remote access and configuration). If the encryption functionality ENCRYPTION_MODE is active, the remote host needs to use the same symmetric MAC_ENCRYPTION_KEY as configured in the client to access configuration as well as SET and GET the pin level.

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 17), or CMD_GPIO_LOCAL_READ_REQ for reading GPIO values (see figure 18).

To control a GPIO via remote device, send the respective commands, CMD_GPIO_REMOTE_WRITE_REQ for setting GPIO output values (see figure 19), or CMD_GPIO_REMOTE_READ_REQ for reading GPIO values (see figure 20).

Each time the GPIOs are written via remote connection, the local host is informed using a CMD_GPIO_REMOTE_WRITE_IND message.

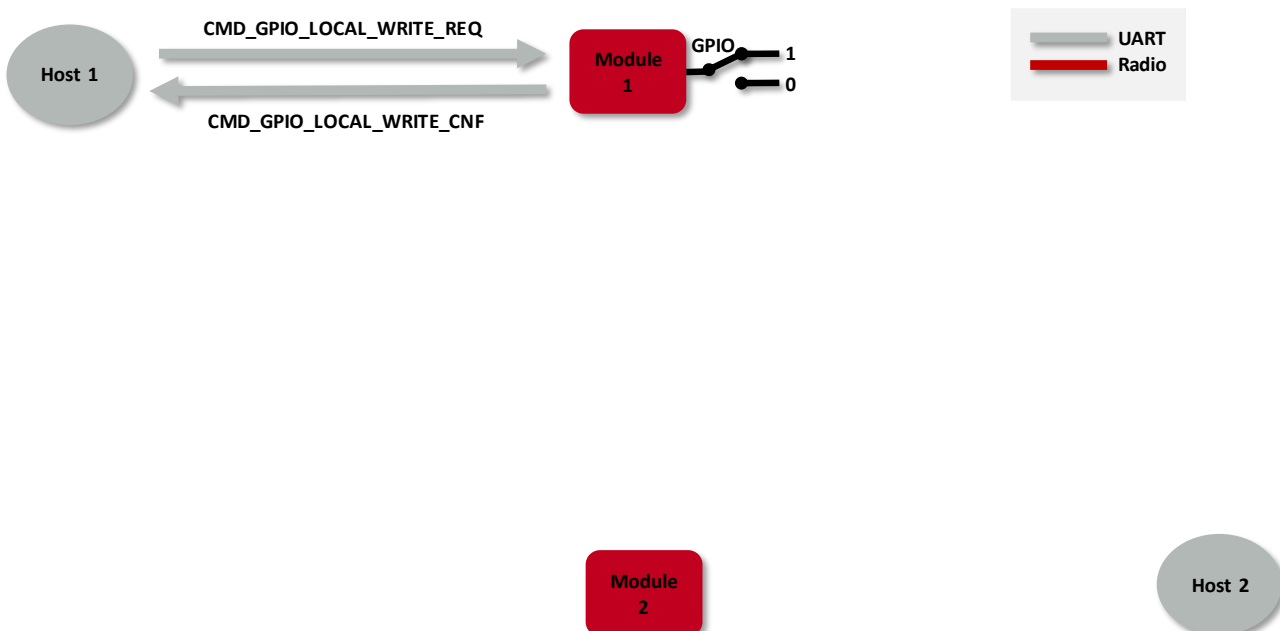


Figure 17: Set the output value of a GPIO via host controller

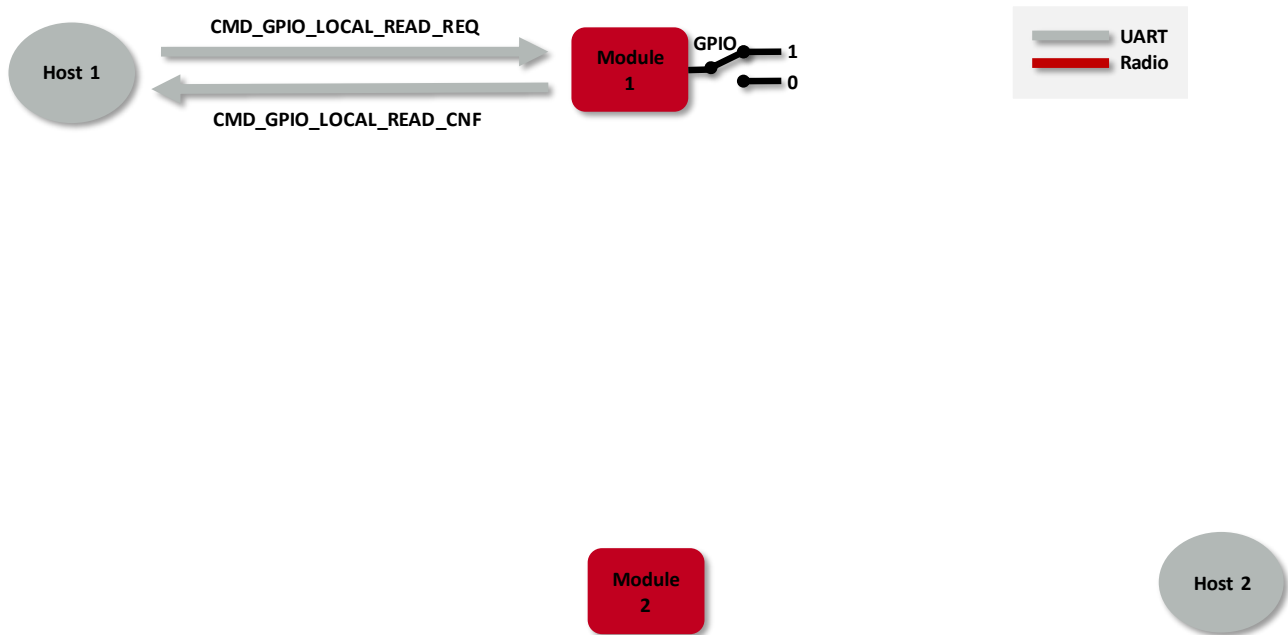


Figure 18: Read the input value of a GPIO via host controller

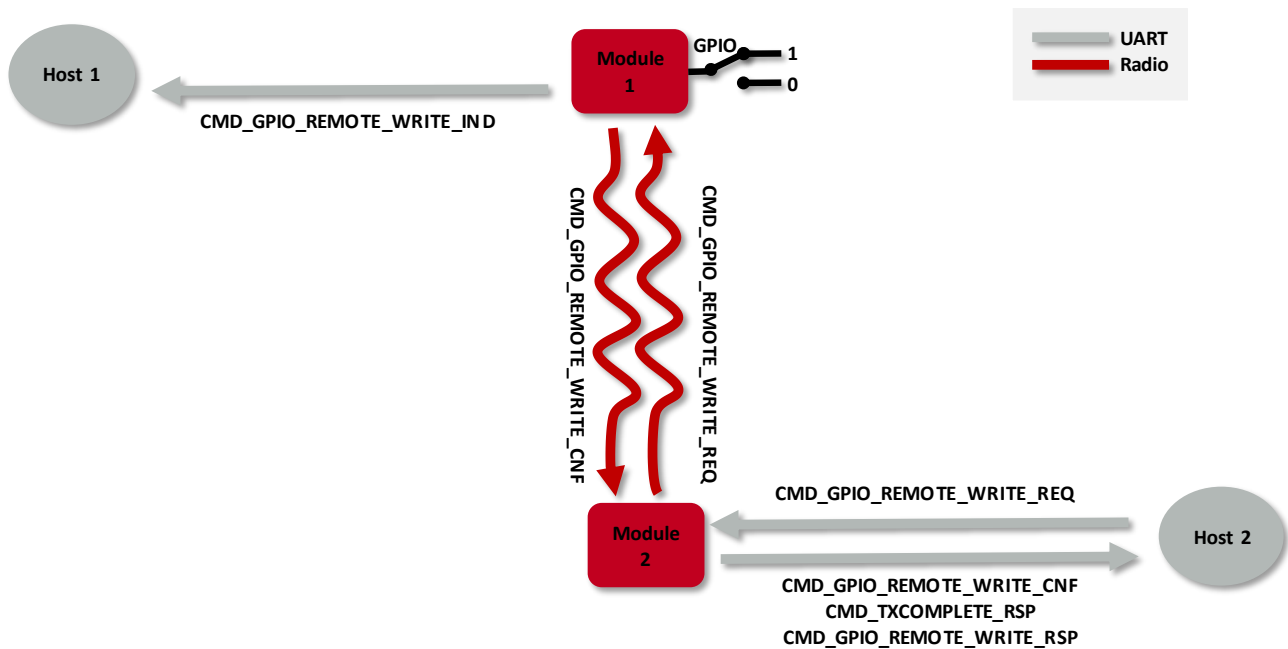


Figure 19: Set the output value of a GPIO via remote device

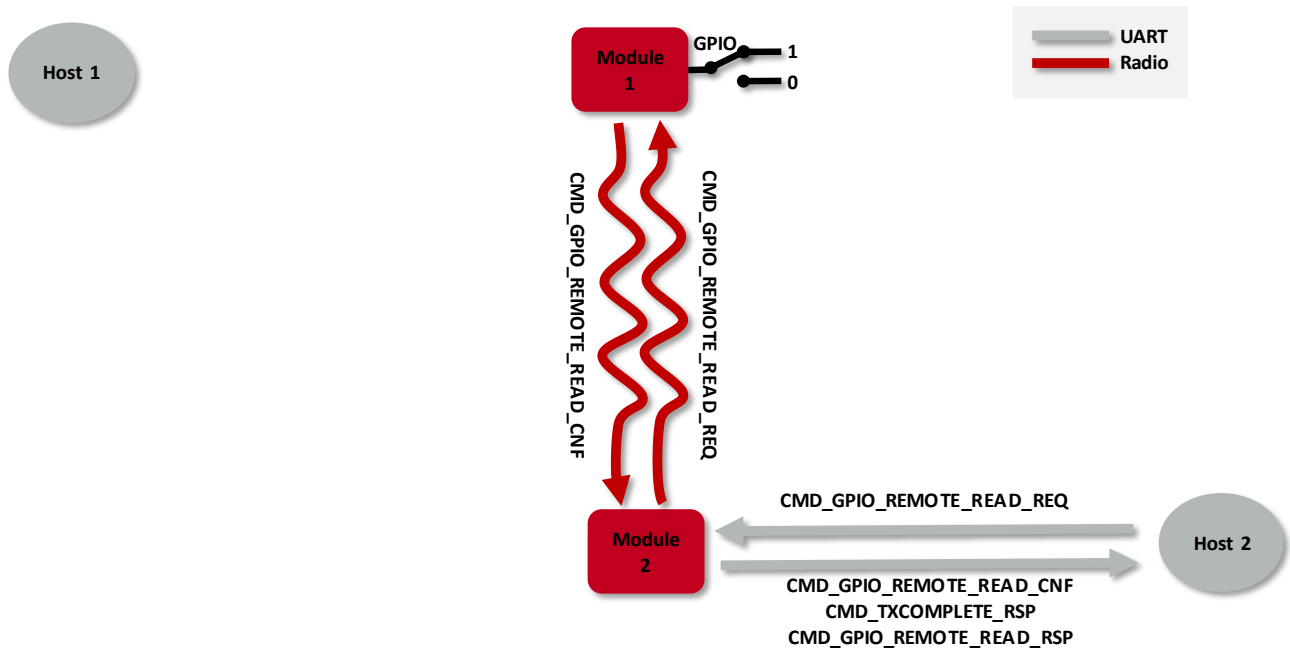
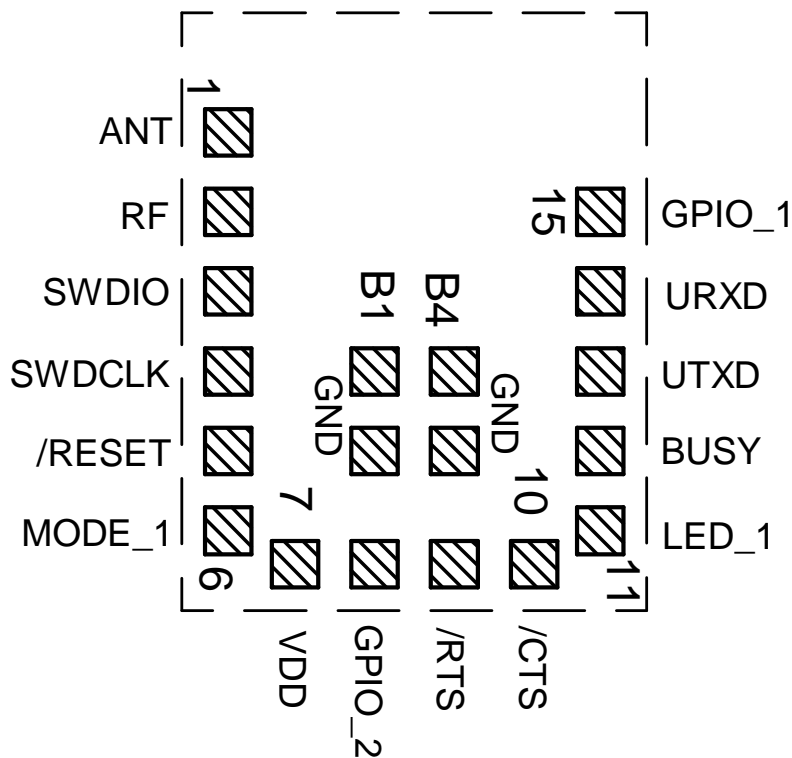


Figure 20: Read the input value of a GPIO via remote device

10.1. Supported GPIOs for remote and local control

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



| Pad No | Name | GPIO_ID | Supported functions |
|--------|--------|---------|--|
| 15 | GPIO_1 | 1 | Input (with and without pull resistor) or output |
| 8 | GPIO_2 | 2 | Input (with and without pull resistor) or output |

Table 24: Supported GPIOs

11. Flooding mesh: Using the repeater functionality

The module can be run as a repeater, to artificially extend the range of sending devices in an existing network.

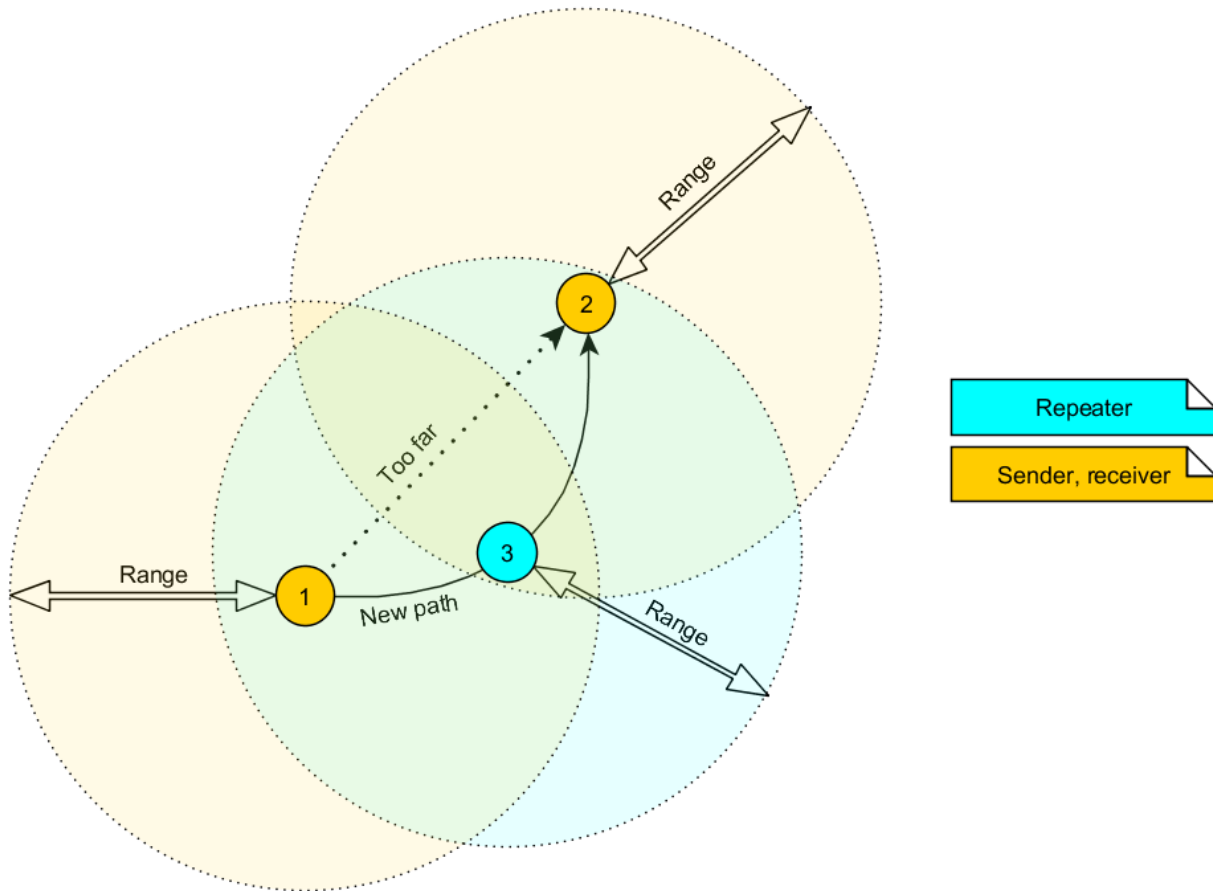


Figure 21: Range extension using several repeaters

If the module is configured as repeater, it can be simply added to an existing wireless network consisting of compatible modules. With this, the newly generated mesh network uses the so-called "flooding technique" to deliver data packets from their source to their destination device.

The repeater module itself simply listens to the configured channel and forwards all received packets except the ones addressed to itself. In case the RSSI value of the received packet is higher than the threshold `RF_REPEATER_THRESHOLD`, the radio packet will not be relayed.

A random delay is used to avoid RF packet collision. To reduce traffic on the frequency channel, each repeater device checks before repetition, if the channel is free and whether it has already sent this packet before or not. Thus, every repeater sends each packet only once. Furthermore, a time-to-live parameter can be set on the radio packet in order to restrict the number of hops.

In a network with N_{umRP} repeater devices, each data packet is repeated N_{umRP} times. Therefore,

each packet that is sent from node A to node B forces a traffic of $N_{\text{umRP}}+1$ data packets in total on the frequency channel.



A module which is configured as repeater, simultaneously also supports also the functions of a standard module. Thus, it can receive data and can initiate the data transmission to other modules.

11.1. Setup of the network and repeater device

The repeater mode can be enabled by setting the `MODULE_MODE` parameter to 2.

If the module is configured as repeater, the following notes have to be considered:

1. Requirements on the network:

- a) The repeater devices have to be line-powered (no battery).
- b) Depending on the data rate, each repeater should repeat a maximum of 2-5 packets per second to give a good chance that the repeater is not busy with repeating when already a new packet arrives for repetition. More packets per second will result in more packet loss, as the collision probability is increased.
- c) If the network consists of several layers of repeaters, each layer adds additional delays to the packet transmission.
- d) To set-up the network, all participants have to use the same `RF_CHANNEL` and `RF_PROFILE`.

2. Information for the repeater device:

- a) Acknowledgements (ACK) of successfully received packets are blocked. If an ACK is requested by the sending module, the request is ignored. Furthermore, the repeater does not request any ACK, when repeating a packet.
- b) The "packet sniffer" mode cannot be run at the same time as the module is in repeater mode.
- c) A time-to-live parameter is present in every packet. The repeater decrements the TTL by one before forwarding the packet. The packet with a TTL value of zero is not repeated.

3. Information for the sending and receiving devices:

- a) The senders should send less frequently to avoid packet collision on the frequency channel.
- b) The repeater devices do not support the feature of ACKs for the successful reception of the packets. Thus, the sender will never receive ACKs, if requested.
- c) Every repeater sends each packet only once. However, receivers can receive each packet several times (sent by different repeaters), if there are packets of different content in the network temporarily close to each other. Thus, on the side of the receiving device, a mechanism that detects and filters double packets shall be implemented.

11.2. Example network

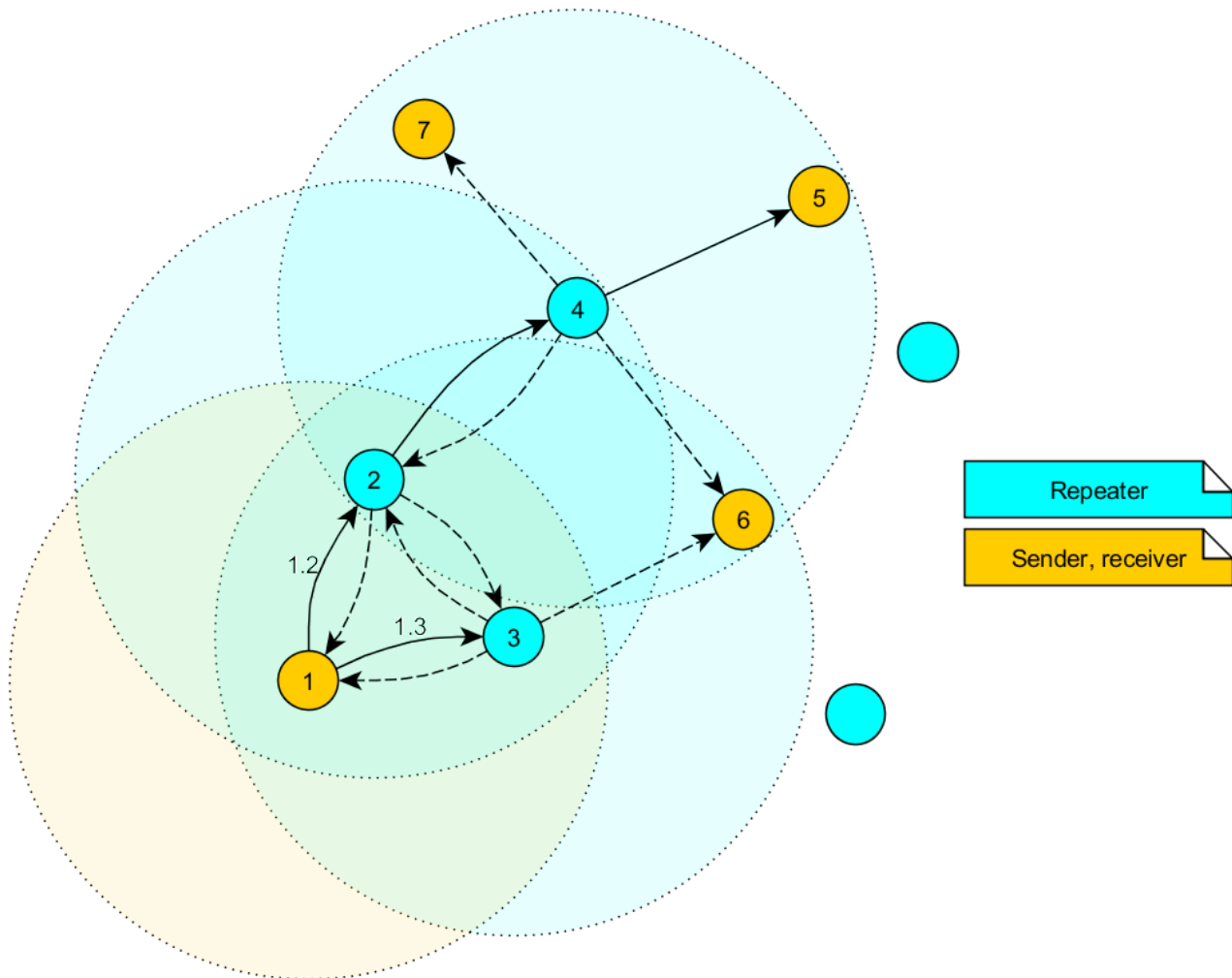


Figure 22: Example network

In the example network shown above, the goal is to send a packet from device 1 to 5. Without the repeater devices, this would be impossible. The steps are as follows:

1. Sender 1 sends a packet.
 - a) Repeater 2 and 3 receive and accept it at the same time.
2. Device 2 and 3 delay the packet.
 - a) Repeater 3 sends the packet.
 - i. Sender 1 and 6 do not accept it, since their addresses are wrong (unequal 5).
 - ii. Repeater 2 does not accept it, since it has been already received before (1.2).
 - b) Repeater 2 sends the packet.
 - i. Repeater 3 does not accept it, since it has been already received before (1.3).
 - ii. Sender 1 does not accept it, since its address is wrong (unequal 5).

- iii. Repeater 4 receives and accepts the packet.
3. Repeater 4 delays and sends the packet.
 - a) Senders 6 and 7 do not accept it, since their addresses are wrong (unequal 5).
 - b) Repeater 2 does not accept it, since it has been already received before (1.2).
 - c) Receiver 5 accepts it and its successfully delivered (address equals 5).

Note that the packet forwarded by repeater 2 and 3 would collide in the frequency channel, if they wouldn't be randomly delayed (see RF_RP_NUM_SLOTS).

11.2.1. Application in parallel networks

As described above, a repeater device forwards all packets that are received before. If a network needs to have a bigger throughput of data, a parallel network can be set up, that relaxes the stress of the primary network. To do so, all sending, receiving and repeater devices of the parallel network are configured to use a new non-overlapping channel, such that the primary network is not affected at all by the traffic of the parallel network.

12. Timing parameters

In this chapter, the timing parameters of the module are listed and described in detail.

12.1. Start-up

After powering the module, the */RESET* pin shall be held LOW for another $\Delta t = t1-t0$ of 1 ms after the *VDD* is stable, to ensure a safe start-up. In the command mode, the module sends a *CMD_START_IND* on the */UTXD* to indicated "ready for operation". This indication is done with *BUSY* pin pulled to LOW in the transparent mode. The timing behaviour is the same in case of wake-up from sleep. Table 25 shows timing parameters for the start-up sequence.



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

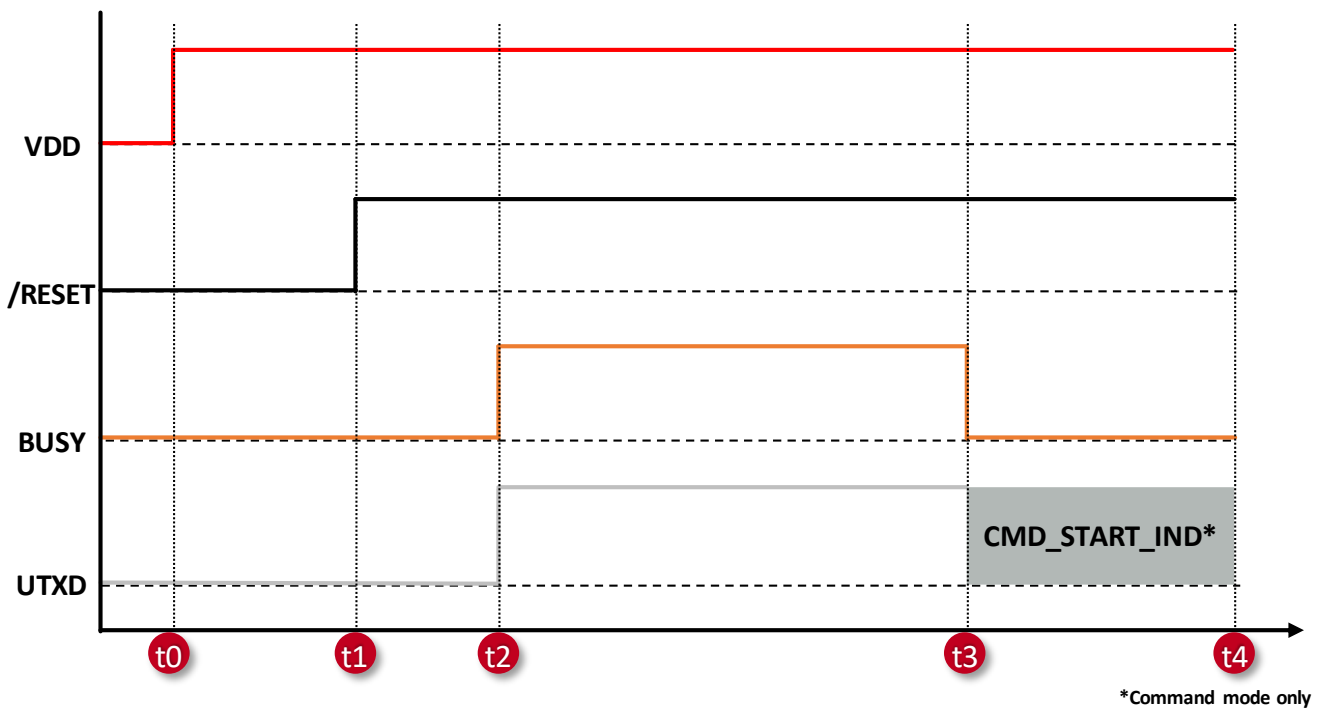


Figure 23: Power up

| Interval | Value [ms] | Description |
|----------|----------------|--|
| t1-t0 | 1 | VDD ramp-up. |
| t2-t1 | 3 | Application pre-initialization. |
| t3-t2 | 1.95 | Application initialization. |
| t4-t3 | X ¹ | CMD_START_IND. |
| t3-t0 | 5.95 | Module ready for operation (transparent mode). |
| t4-t0 | 5.95 + X | Module ready for operation(command mode). |

Table 25: Start up timing

¹Time taken to send 8 bytes of the start indication packet in the configured baud rate.

12.2. Data transmission and throughput measurements

12.2.1. Command mode

In the command mode, the host micro-controller sends a data transmit request with one of the following commands: `CMD_UNICAST_DATA_REQ`, `CMD_MULTICAST_DATA_REQ`, `CMD_BROADCAST_DATA_REQ`, `CMD_UNICAST_DATA_EX_REQ` or `CMD_MULTICAST_DATA_EX_REQ`. The module buffers the data and acknowledges the same over UART with a `CMD_DATA_CNF` message. A data transmit is triggered on completion of which the host receives a `CMD_TXCOMPLETE_RSP`. The figure 24 shows the sequence of data transmission in this mode.

The table 26 lists the timings for throughput measurement done with an STM32 Nucleo-L073RZ micro-controller as host with a UART baud rate of 460800 Baud.

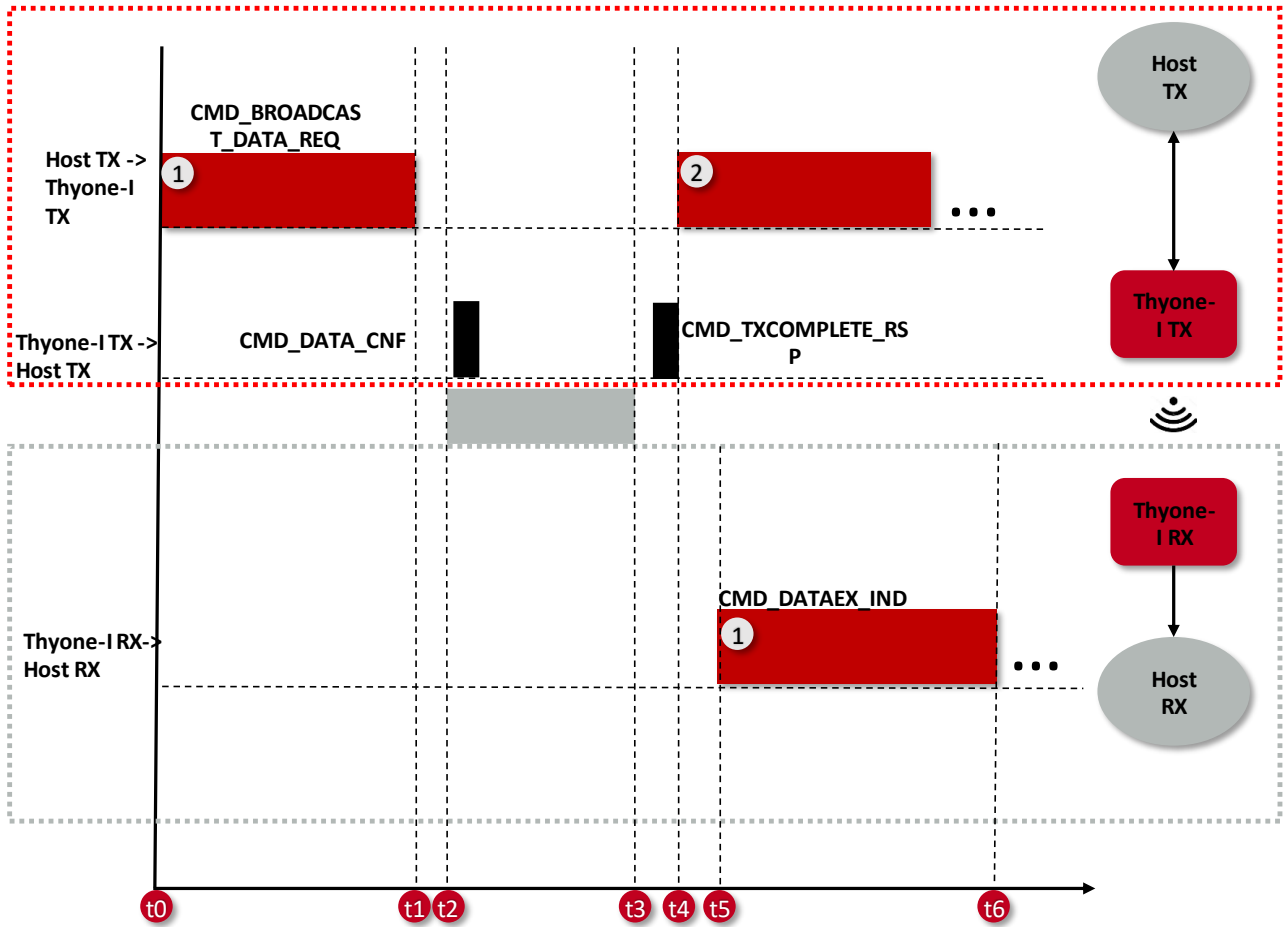


Figure 24: Command sequence when transmitting data

| 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) | Throughput (8×224/(t4-t0)) [kBit/s] |
|------------|-------------------------|--------------------------------|-------------------------|----------------------------------|---|
| 1 Mb/s | 4.92 | 7.47 | 4.97 | 12.48 | 239 |
| 2 Mb/s | 4.92 | 6.5 | 4.97 | 11.51 | 275 |

Table 26: Maximum throughput timings, packet error rate = 0%

12.2.2. Transparent mode

In the transparent mode, the module acts as a bridge forwarding all the data received on the UART over the radio. The figure 25 shows the sequence of data transmission in this mode. The table 27 lists the timings for throughput measurement done with an STM32 Nucleo-L073RZ micro-controller as host with a UART baud rate of 460800 Baud.

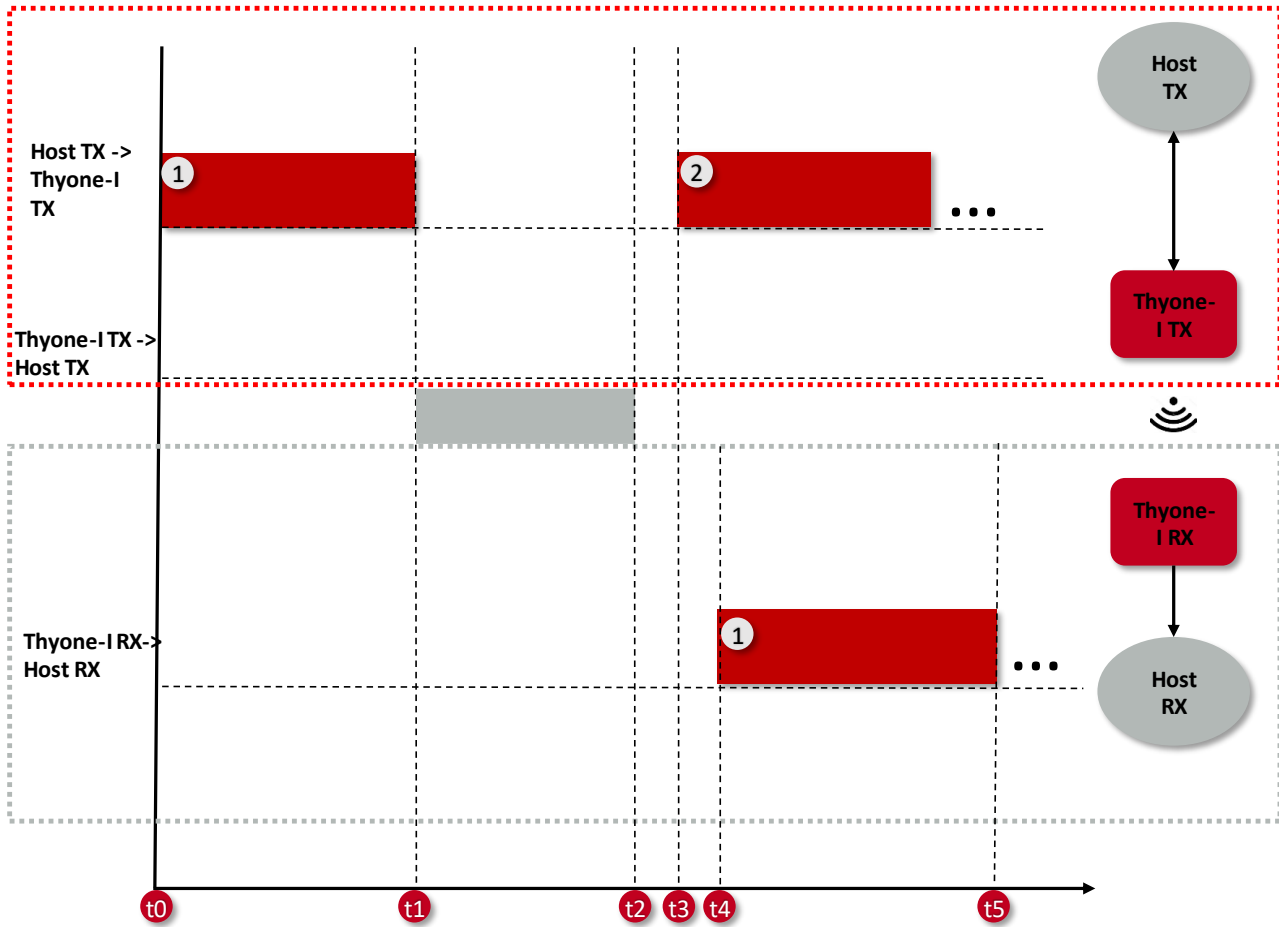


Figure 25: Transmitting data in transparent mode

| Radio mode | $t_1 - t_0$ [ms] (UART RX) | $t_3 - t_0$ [ms] (Host TX Period) | $t_5 - t_4$ [ms] (UART TX) | $t_5 - t_0$ [ms] (End-to-end delay) | Throughput ($8 \times 224 / (t_3 - t_0)$) [kBit/s] |
|------------|-------------------------------|--------------------------------------|-------------------------------|--|--|
| 1 Mb/s | 4.76 | 7.03 | 4.8 | 12.34 | 254 |
| 2 Mb/s | 4.76 | 6.06 | 4.8 | 11.4 | 295 |

Table 27: Maximum throughput timings, packet error rate = 0%

13. Custom firmware

13.1. Custom configuration of standard firmware

The configuration of the standard firmware includes adoption of the non-volatile user settings (see chapter 9) 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.

13.2. 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 such 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.

13.3. 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 a 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 firmware cannot be applied to this customer firmware solution without a review and verification.

13.4. Contact for firmware requests

Please contact your Business Development Engineer (BDM) or WCS@we-online.com for quotes regarding these topics.

14. Firmware history

14.1. Release notes

Version 1.9.0 "Release"

- Initial release

15. Design in guide

15.1. 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.



No fixed values can be recommended, as these depend on the circumstances of the application (main power source, interferences etc.).



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.

- 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.



Again, no fixed values can be recommended, as they depend on the influencing circumstances of the application (antenna, interferences etc.).

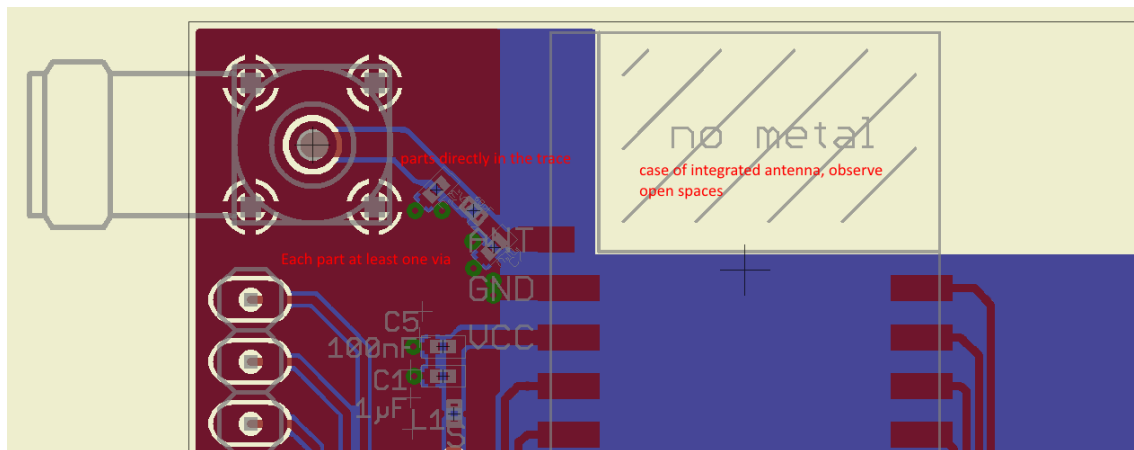


Figure 26: Layout

- 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.
- 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.

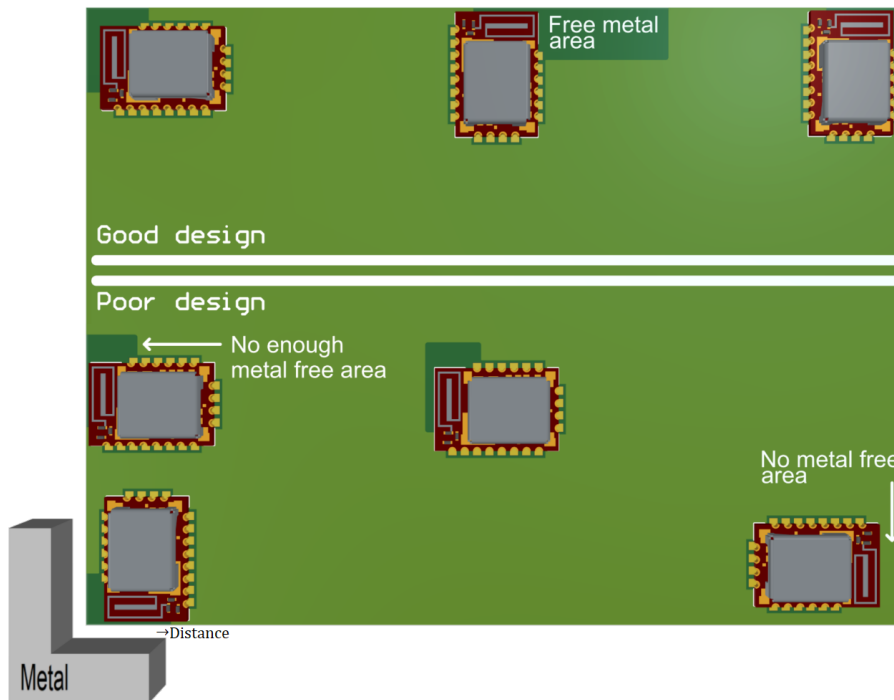


Figure 27: Placement of the module with integrated antenna

15.2. Designing the antenna connection

The antenna should be connected with a 50 Ω 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 Ω 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.

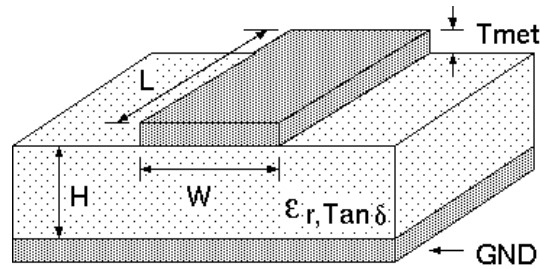


Figure 28: Dimensioning the antenna connection as micro strip

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

$$W = 1.25 \times \left(\frac{5.98 \times H}{e^{\frac{50 \times \sqrt{\epsilon_r + 1.41}}{87}}} - T_{met} \right) \quad (1)$$

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.

15.3. 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 electro-magnetic field blocking.

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

15.3.1. 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 40Ω input impedance. Therefore, matching is not required.

15.3.2. 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.

15.3.3. 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.

15.3.4. Antennas provided by Würth Elektronik eiSos

15.3.4.1. 2600130021 - Himalia dipole antenna



Figure 29: 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.

16. Reference design

Thyone-e was tested and certified on the corresponding Thyone-e evaluation board. For the European Conformity, the evaluation board serves as a 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 Thyone-e evaluation board.

16.1. EV-Board

16.1.1. Schematic

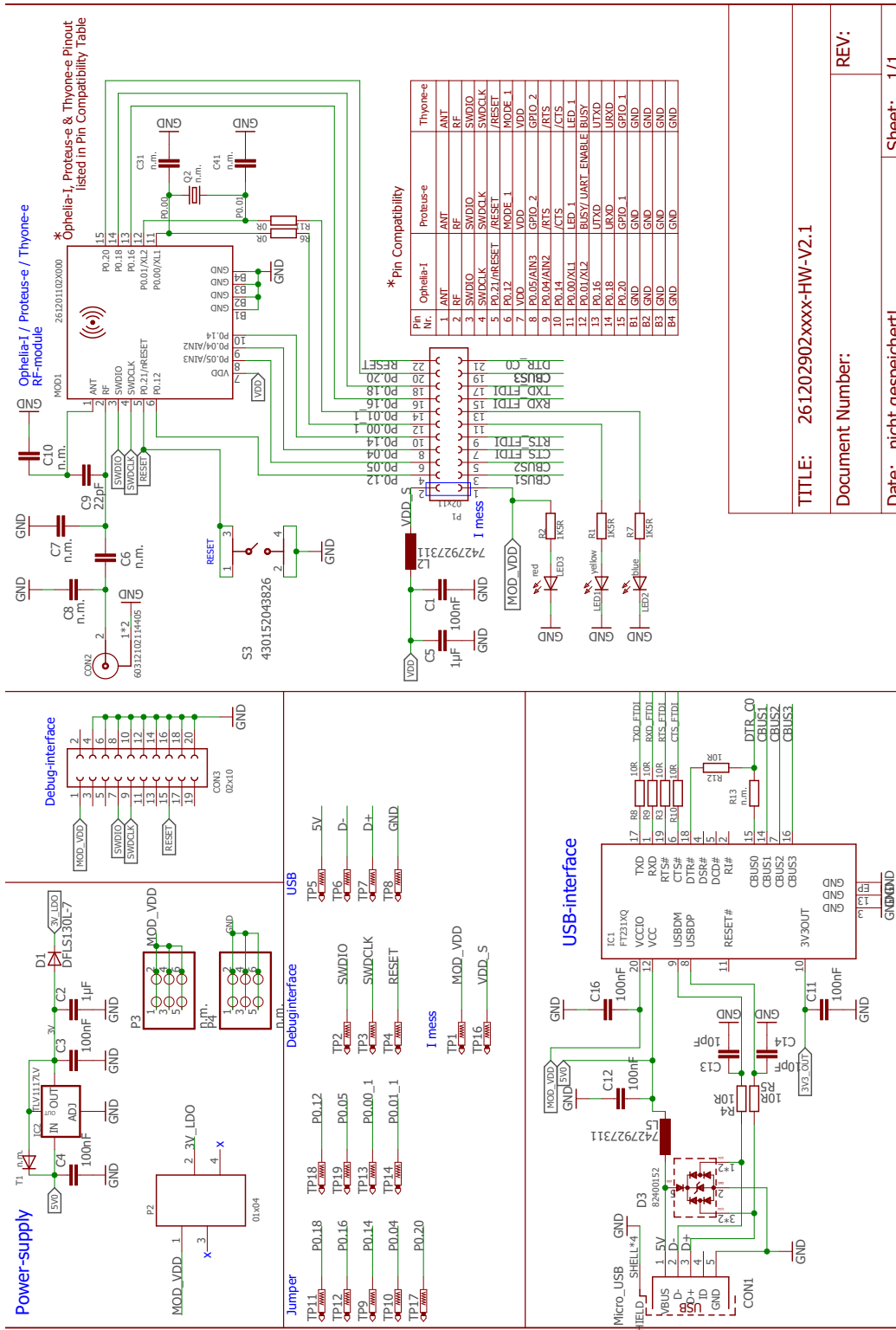


Figure 30: Reference design: Schematic

16.2. Layout

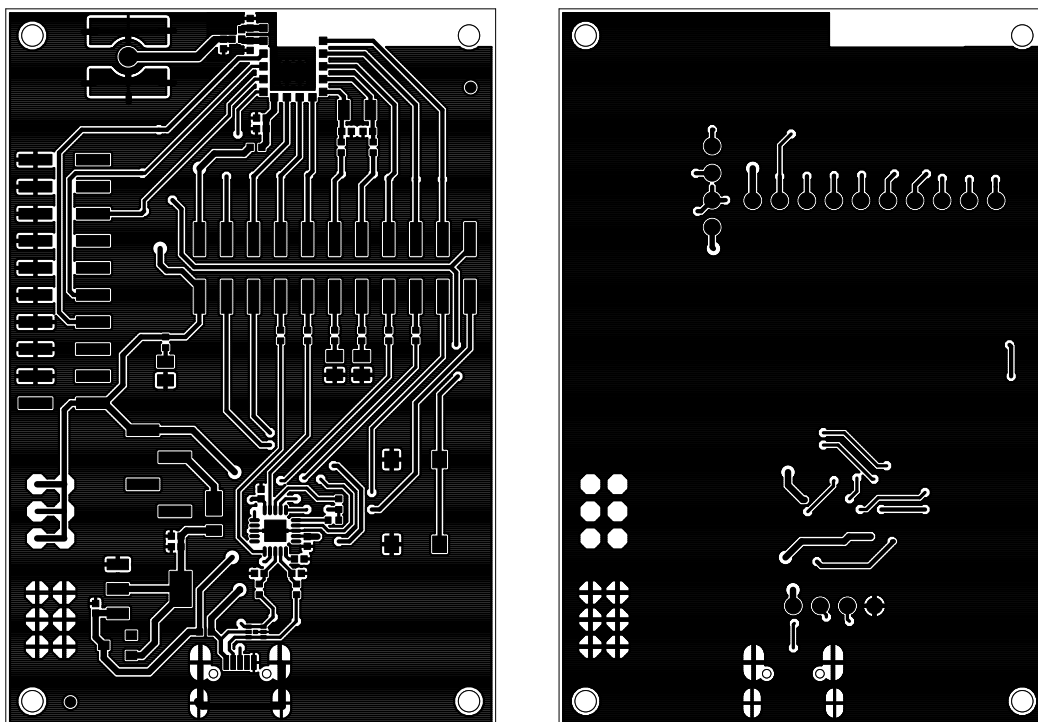


Figure 31: Top layer (top), bottom layer (bottom)

16.3. Internal antenna radiation characteristics

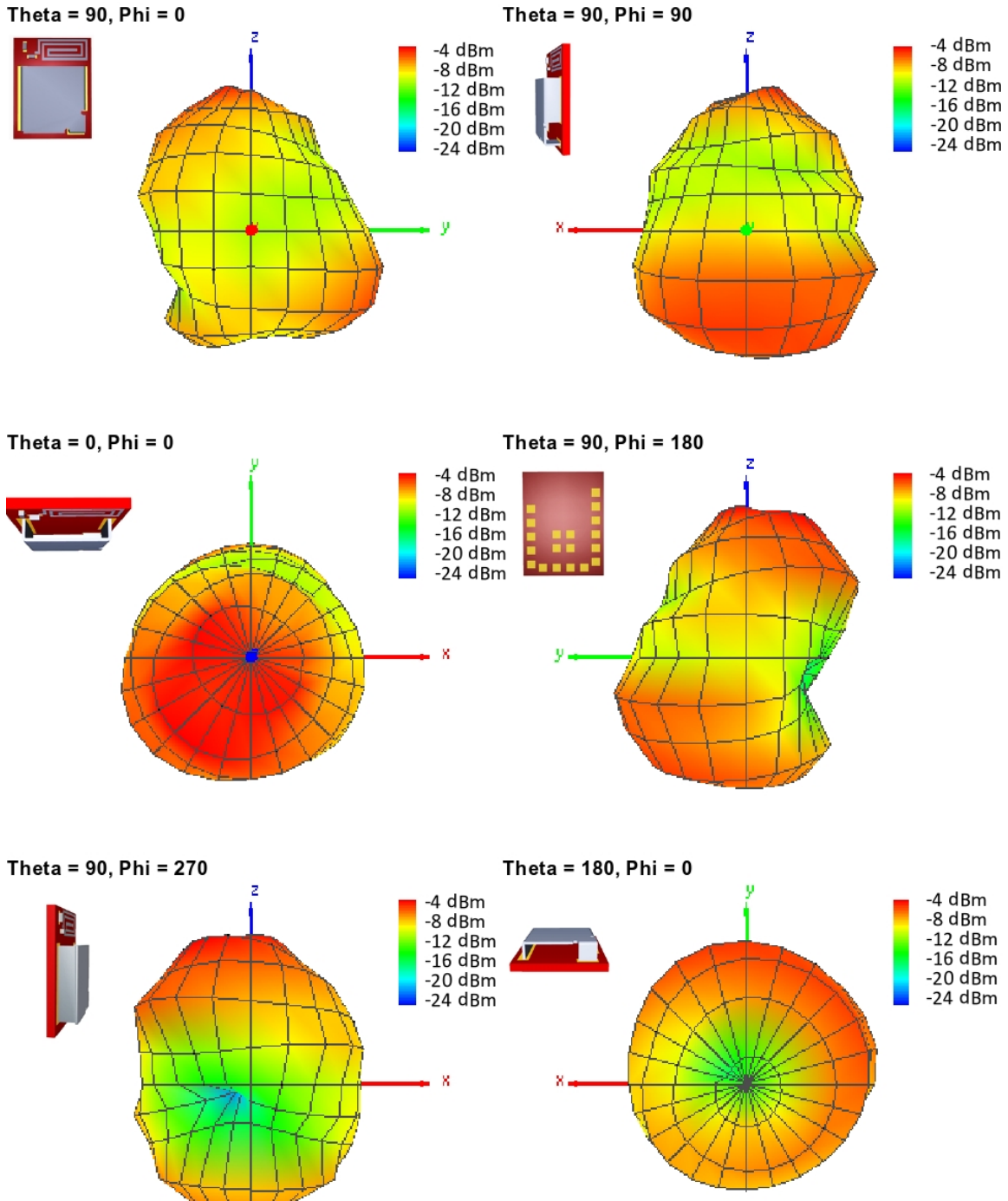


Figure 32: Antenna characteristic from integrated antenna measured on official evaluation board*

*Radiation characteristic shown is valid for the module on the evaluation board. It is important to be aware that size of ground plane and placement of module has influence on the radiation pattern.

16.4. Trace design

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



To reuse and refer 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 is to be used, if size and price is critical and the range is uncritical. See chapter 16.4.1.
- A 22 pF capacitor, connecting *RF* and *ANT* pin is to be used, if size and price is less critical, but an assembly variant with external antenna is also being used. See chapter 16.4.2.
- A 22 pF capacitor, connecting *RF* pin to the external pre-certified dipole antenna (Himalia [4], chapter 15.3.4.1). This configuration suits best if size and price is less critical, but radio range should be optimized. See chapter 16.4.3.

The trace designs use the same layer stack up.

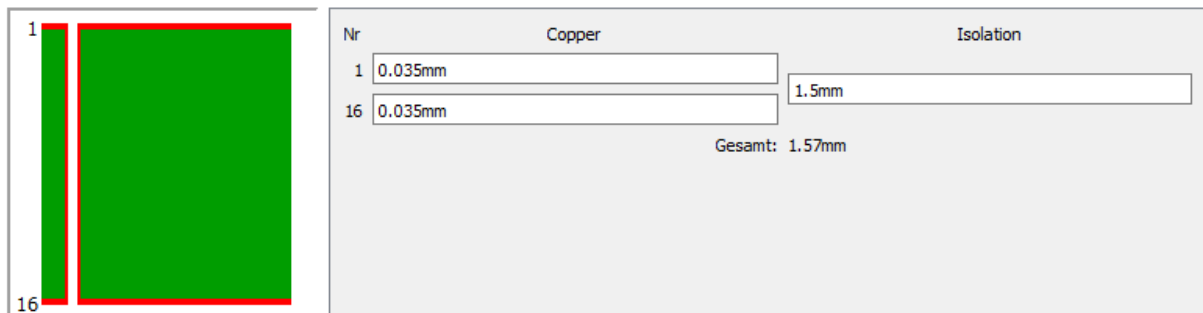


Figure 33: Stack-up

- 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.

16.4.1. Simple short using internal antenna

The simple short is a 50 Ω coplanar strip connecting *RF* and *ANT* pin. Figures 34 and 35 show this in detail.

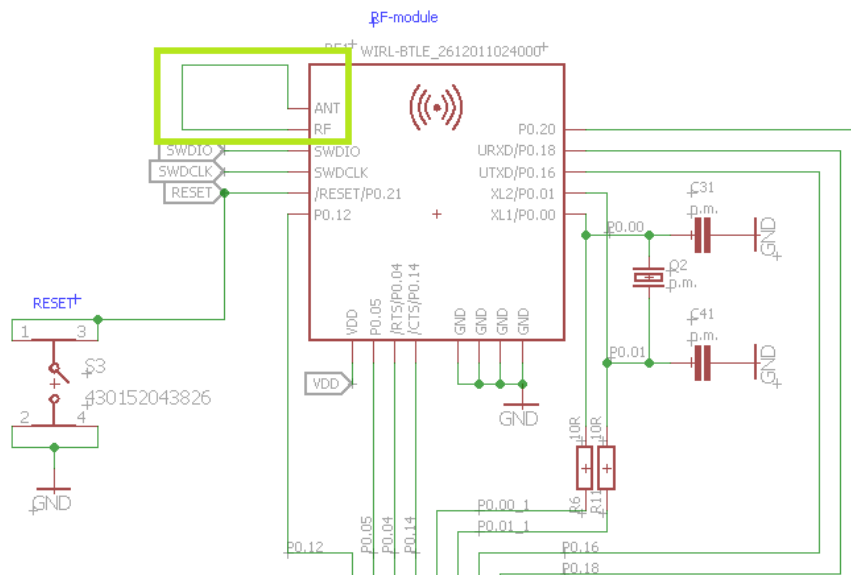


Figure 34: Simple short schematic

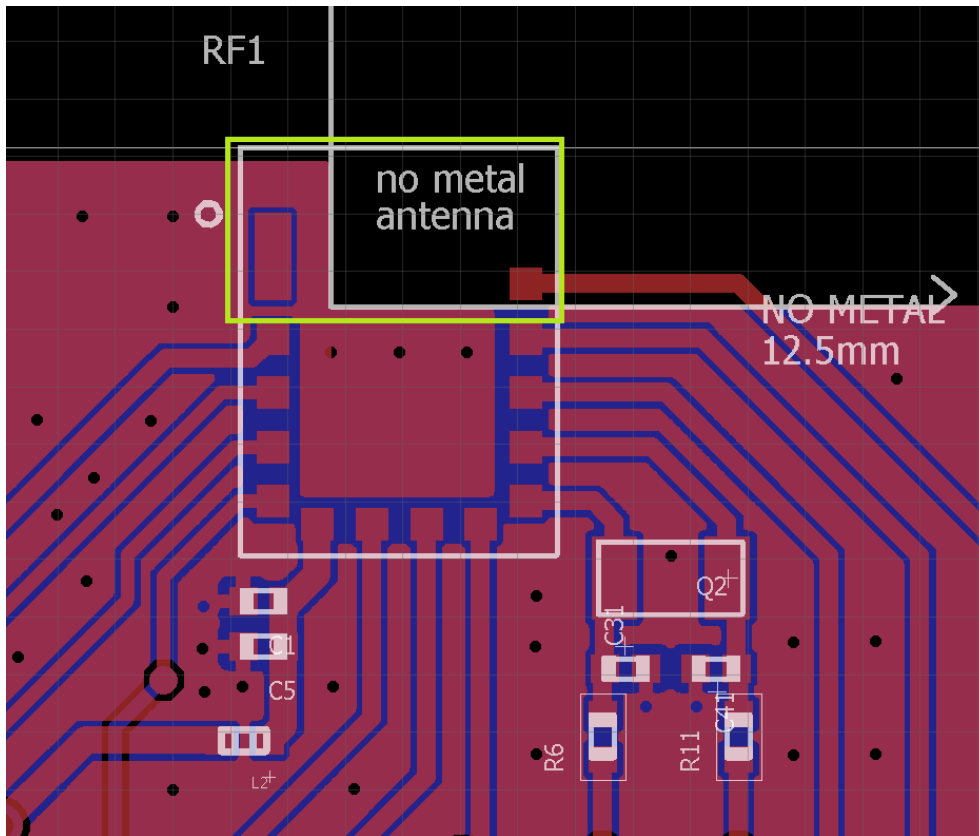


Figure 35: Simple short layout

16.4.2. 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 36 and 37 show this in detail.

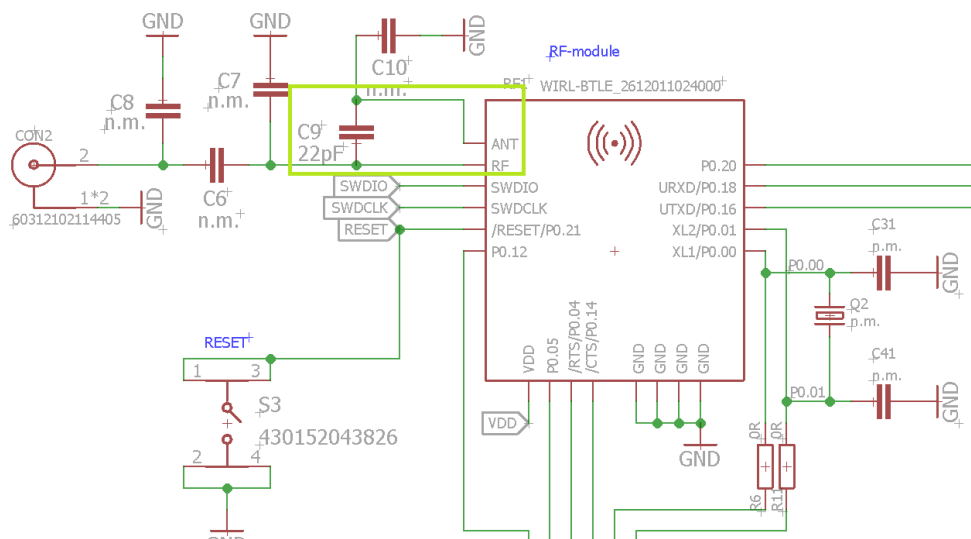


Figure 36: Capacitor internal antenna schematic

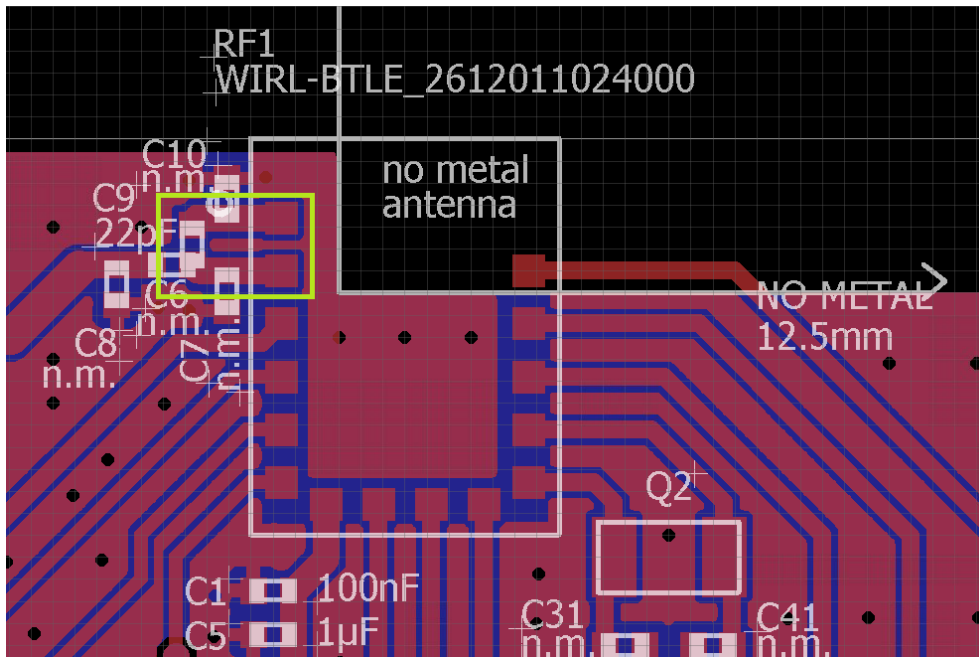


Figure 37: Capacitor internal antenna layout

16.4.3. 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 [4], chapter 15.3.4.1). C7, C8, C9 and C10 are left unassembled. Figures 38 and 39 show this in detail.

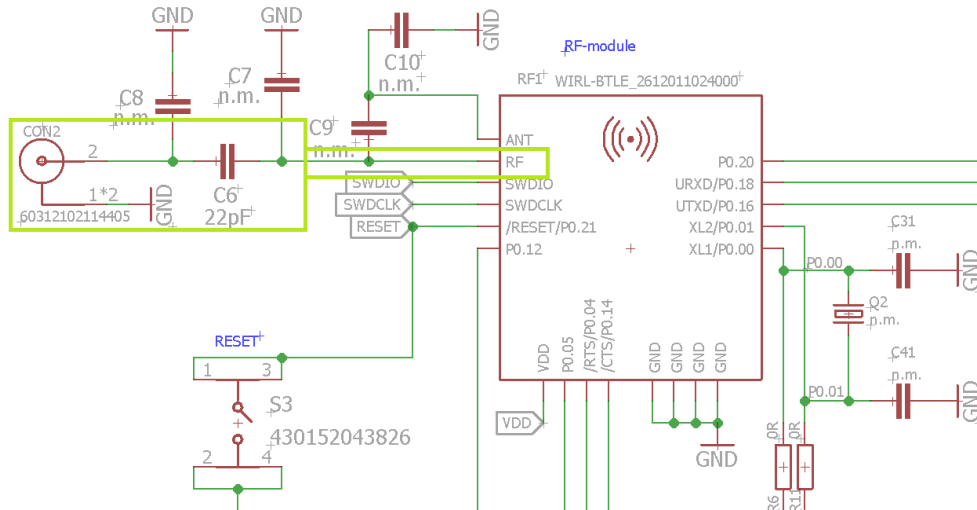


Figure 38: Capacitor external antenna schematic

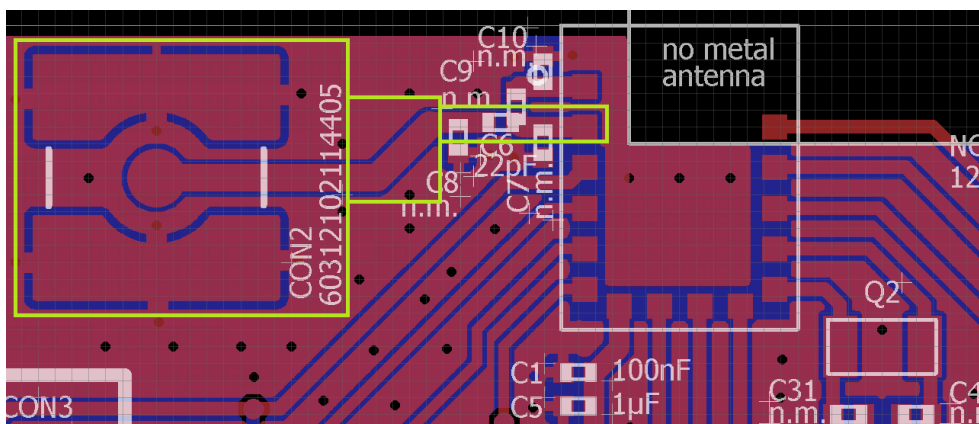


Figure 39: Capacitor external antenna layout



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.

16.5. 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 36.

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 16.4.2 as implemented on the evaluation board gives the possibility to either follow the trace design or do fine tuning if needed without changing the PCB design.

17. Manufacturing information

17.1. 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.

17.2. Soldering

17.2.1. 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 seconds |
| Ramp-up rate (T_L to T_P) | | 3 °C / second max. |
| Liquidous temperature | T_L | 217 °C |
| Time t_L maintained above T_L | t_L | 60 - 150 seconds |
| Peak package body temperature | T_P | 260 °C |
| Time within 5 °C of actual peak temperature | t_P | 20 - 30 seconds |
| Ramp-down Rate (T_P to T_L) | | 6 °C / second max. |
| Time 20 °C to T_P | | 8 minutes 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.

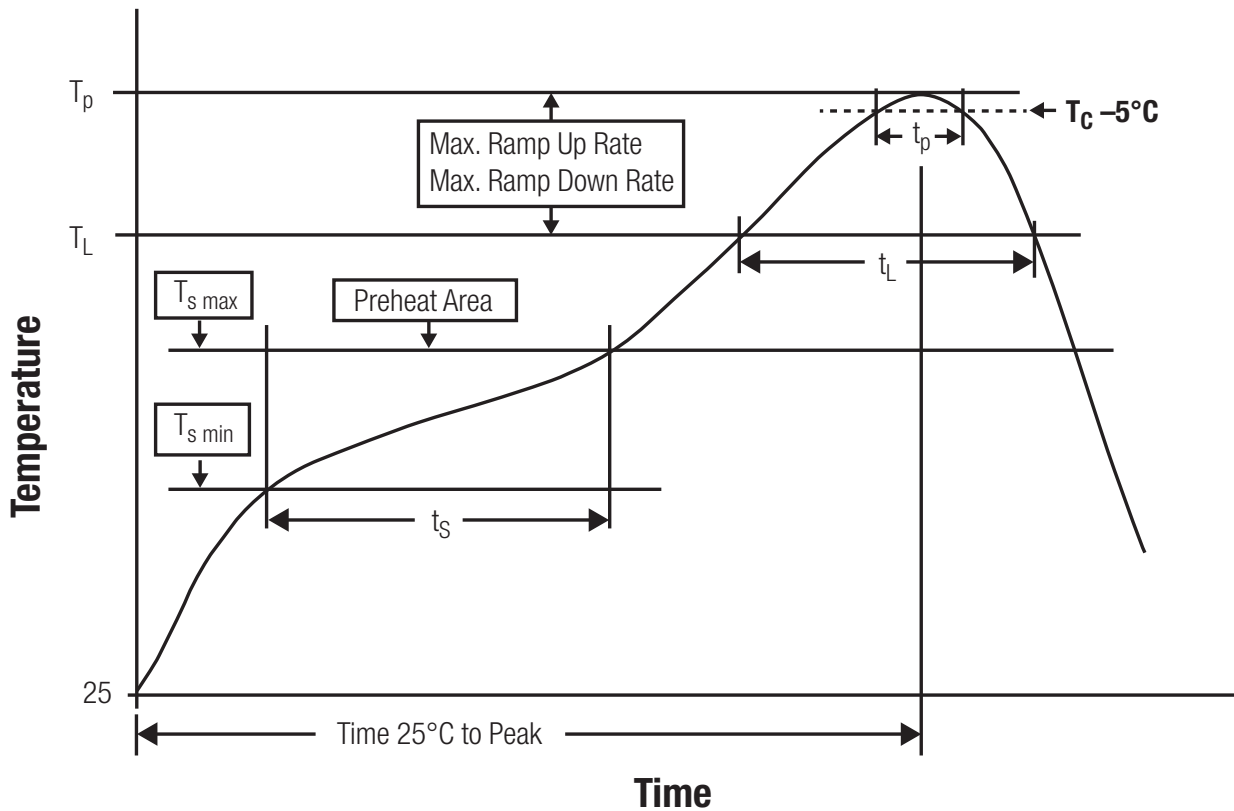


Figure 40: Reflow soldering profile

After reflow soldering, visually inspect the board to confirm proper alignment

17.2.2. 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.

17.2.3. 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 behaviour of the device. This might be critical for components with high power.

17.2.4. 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.

17.3. 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 2. 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.

17.4. 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.

18. Physical specifications

18.1. Dimensions

| Dimensions |
|--------------|
| 9 x 7 x 2 mm |

Table 29: Dimensions

18.2. Weight

| Weight |
|--------|
| < 1 g |

Table 30: Weight

18.3. Module drawing

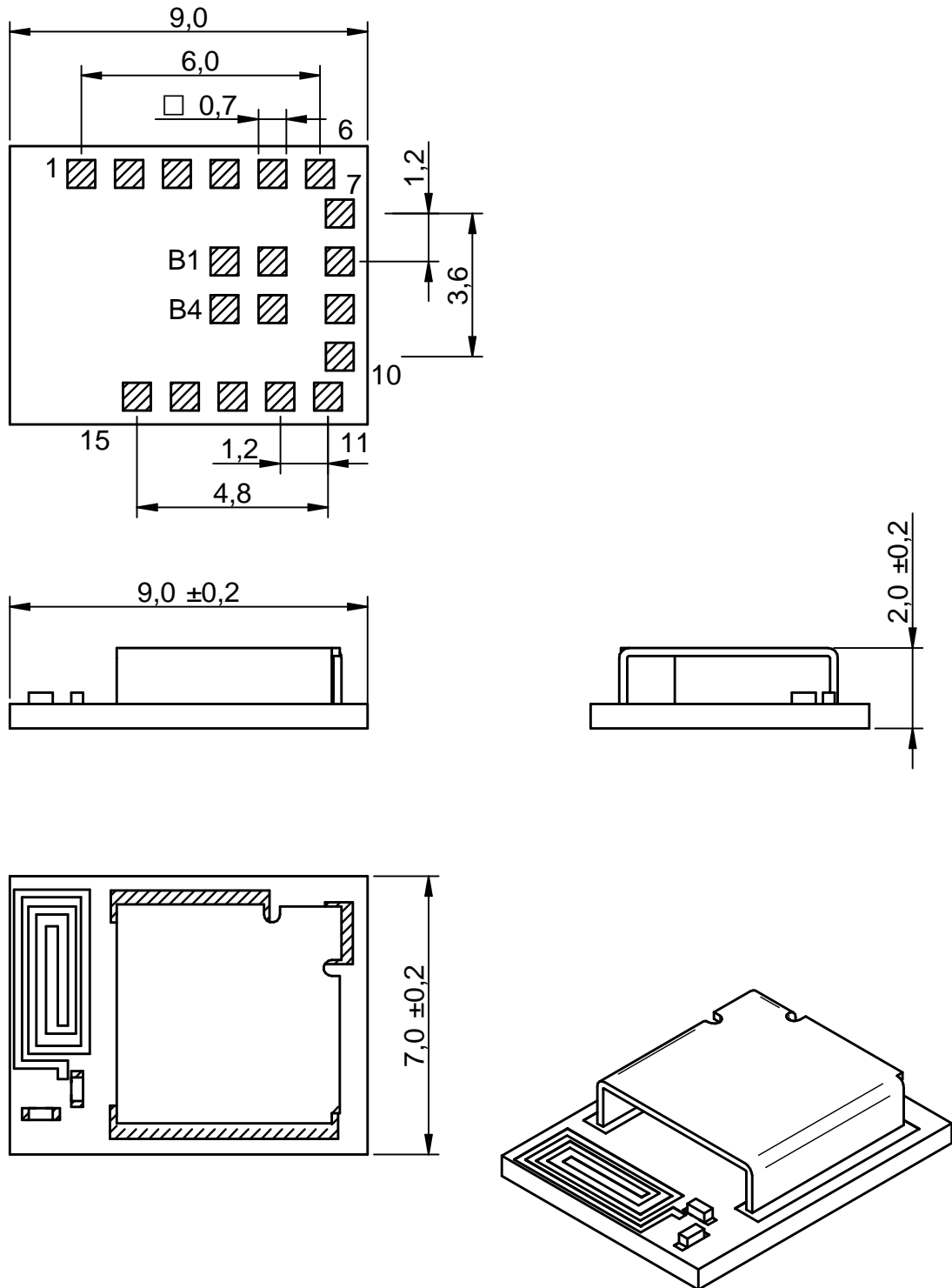


Figure 41: Module dimensions [mm]

18.4. Footprint WE-FP-4+

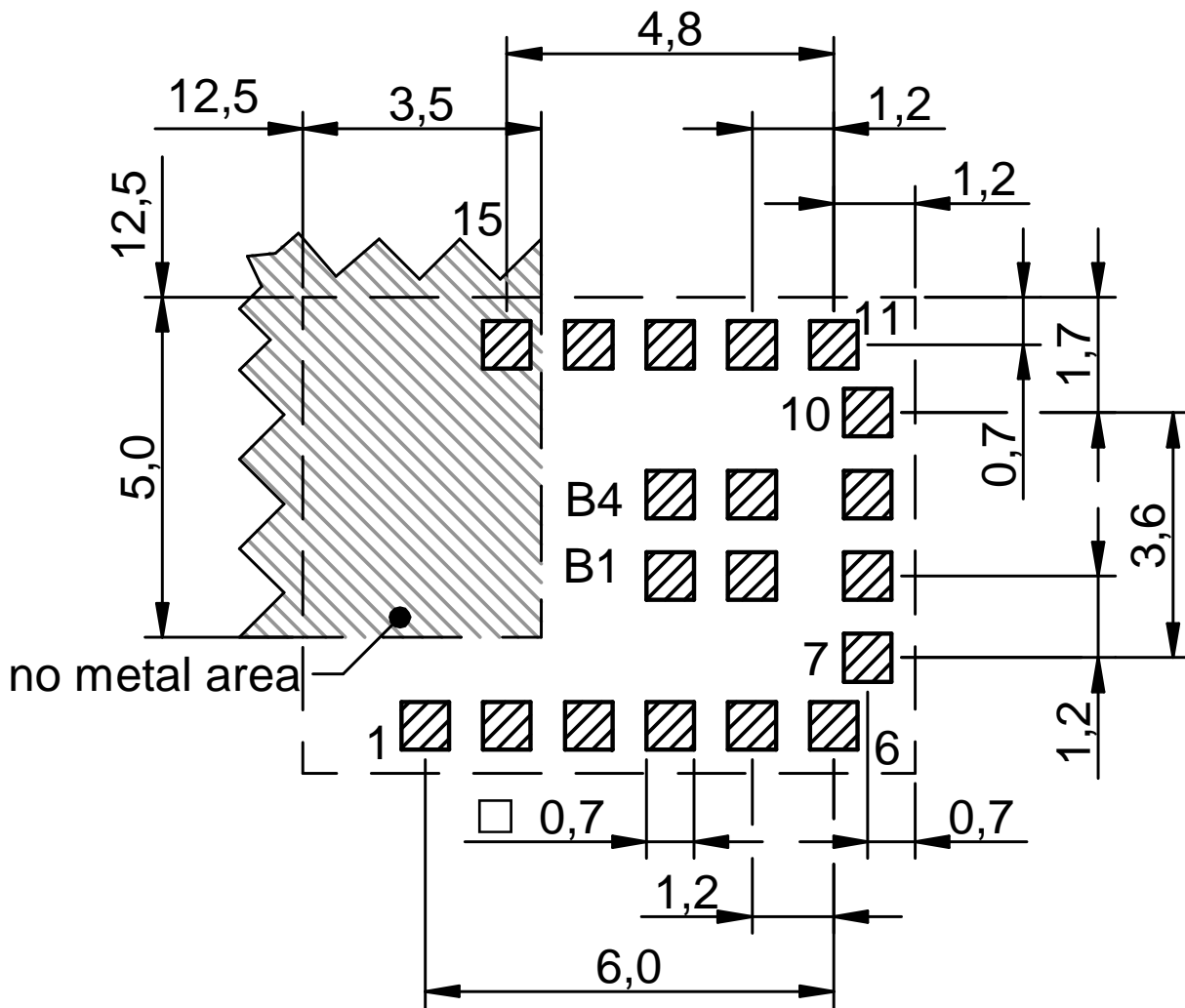


Figure 42: Footprint WE-FP-4+ [mm]

18.5. 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 42). Even though metal parts would influence the characteristic of the antenna, but the direct influence and matching keep an acceptable level.

19. Marking

19.1. 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.

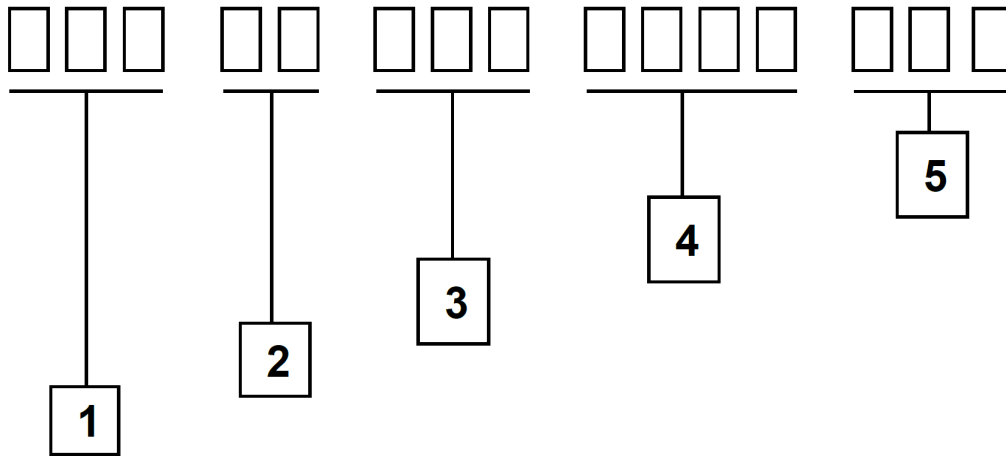


Figure 43: Lot number structure

| 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.

19.2. 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 Thyone-e the label is as follows:

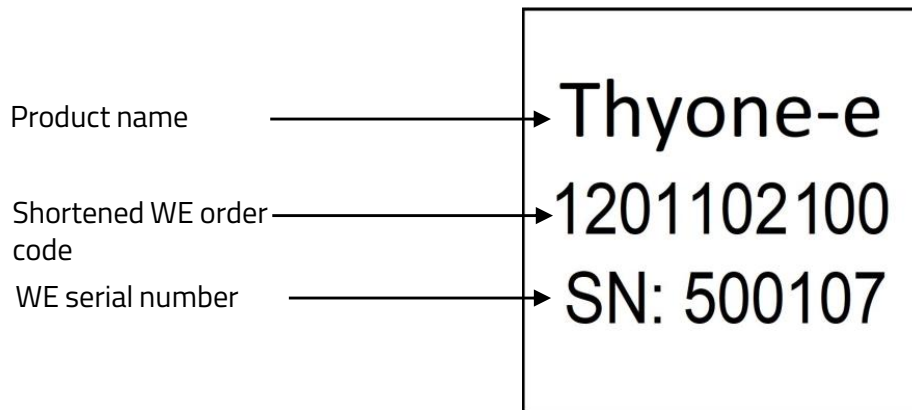


Figure 44: Label of the Thyone-e

20. 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.

21. References

- [1] FTDI virtual COM port driver. <https://ftdichip.com/drivers/vcp-drivers/>.
- [2] Würth Elektronik. WE UART Terminal PC tool (Smart Commander). <https://www.we-online.com/SmartCommander>.
- [3] Würth Elektronik. Wireless Connectivity SDK for STM32 - Radio module drivers in C-code. https://github.com/WurthElektronik/WirelessConnectivity-SDK_STM32.
- [4] Würth Elektronik. Himalia. https://www.we-online.com/catalog/en/WIRL_ACCE_2600130021.

22. Regulatory compliance information

22.1. Important notice EU

The use of RF frequencies is limited by national regulations. The Thyone-e has been designed to comply with the RED directive 2014/53/EU of the European Union (EU).

The Thyone-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.



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.

22.2. Important notice FCC

The use of RF frequencies is limited by national regulations. The Thyone-e has been designed to comply with the FCC Part 15.

The Thyone-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.

22.3. Conformity assessment of the final product

The Thyone-e is a subassembly. It is designed to be embedded into other products (products incorporating the Thyone-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 Thyone-e carried out by Würth Elektronik eiSos does not replace the required conformity assessment of the final product.

22.4. 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.

22.5. EU Declaration of conformity



EU DECLARATION OF CONFORMITY

Radio equipment: 2612011021000

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: 2612011021000

The object of the declaration described above is in conformity with the relevant Union harmonisation legislation Directive 2014/53/EU and 2011/65/EU with its amending Annex II EU 2015/863 . 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 V3.2.4 (2020-09)
EN 62479 : 2010
EN 62368-1:2014 + AC:2015

i.A. G. Exlerdt

Trier, 9th of September 2024

Place and date of issue

22.6. 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.

22.7. 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.

22.8. 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.
- To comply with FCC/IC regulations limiting both maximum RF output power and human exposure to RF radiation, the maximum antenna gain including cable loss in a mobile-only exposure condition must not exceed 6dBi.
- A label must be affixed to the outside of the host product with the following statements:
This device contains FCCID: R7T1201102
This equipment contains equipment certified under ICID: 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.

22.8.1. OEM requirements:

The OEM must ensure that the following conditions are met.

- The Thyone-e will be used at a distance of at least 10 mm.
- 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 OEM must sign the OEM Modular Approval Agreement.
 - The module must be used with only the following approved antenna(s).

22.8.2. Pre-certified antennas

The Thyone-e is pre-certified with the following antennas.

| Product | Certified antenna |
|--------------------------|--------------------------------------|
| Thyone-e (2612011021000) | PCB antenna included in the Thyone-e |

| | | | |
|----|--|--|------------------------------|
| 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 |
| | CTC advanced GmbH & Untertkheimer Strae 6 10 66117 Saarbrcken/GERMANY | D-PL-12076-01-03 | 1-1754/21-04-03 & 08-11-2021 |

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 46: ETA-WPC certificate page 2

23. 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 23 and 23 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.

24. 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.

25. 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. | Thyone-e | 10 |
| 2. | Block diagram of the module | 12 |
| 3. | Sleep current (no RAM retention, wake on reset) over operating temperature range | 15 |
| 4. | Pinout (top view) | 18 |
| 5. | Minimal pin connections | 20 |
| 6. | Set-up quick start | 24 |
| 7. | Module A command sequence | 26 |
| 8. | Module B command sequence | 27 |
| 9. | Overview modes | 33 |
| 10. | Host interface | 34 |
| 11. | Transparent mode escape sequence timing | 76 |
| 12. | Transparent mode flow control | 77 |
| 13. | Configure the local GPIOs via local host | 111 |
| 14. | Configure the local GPIOs via remote device host | 112 |
| 15. | Read the configuration of the local GPIOs via local host | 113 |
| 16. | Read the configuration of the local GPIOs via remote device host | 113 |
| 17. | Set the output value of a GPIO via host controller | 114 |
| 18. | Read the input value of a GPIO via host controller | 115 |
| 19. | Set the output value of a GPIO via remote device | 115 |
| 20. | Read the input value of a GPIO via remote device | 116 |
| 21. | Range extension using several repeaters | 118 |
| 22. | Example network | 120 |
| 23. | Power up | 122 |
| 24. | Command sequence when transmitting data | 125 |
| 25. | Transmitting data in transparent mode | 126 |
| 26. | Layout | 131 |
| 27. | Placement of the module with integrated antenna | 132 |
| 28. | Dimensioning the antenna connection as micro strip | 133 |
| 29. | Himalia dipole antenna | 135 |
| 30. | Reference design: Schematic | 137 |
| 31. | Top layer (top), bottom layer (bottom) | 138 |
| 32. | Antenna characteristic from integrated antenna measured on official evaluation board* | 139 |
| 33. | Stack-up | 140 |
| 34. | Simple short schematic | 141 |
| 35. | Simple short layout | 142 |
| 36. | Capacitor internal antenna schematic | 142 |
| 37. | Capacitor internal antenna layout | 143 |
| 38. | Capacitor external antenna schematic | 144 |
| 39. | Capacitor external antenna layout | 144 |
| 40. | Reflow soldering profile | 147 |
| 41. | Module dimensions [mm] | 151 |
| 42. | Footprint WE-FP-4+ [mm] | 152 |
| 43. | Lot number structure | 153 |
| 44. | Label of the Thyone-e | 154 |
| 45. | ETA-WPC certificate page 1 | 162 |

| | |
|--|-----|
| 46. ETA-WPC certificate page 2 | 163 |
|--|-----|

List of Tables

| | |
|--|-----|
| 1. Ordering information | 12 |
| 2. Recommended operating conditions | 13 |
| 3. Absolute maximum ratings | 13 |
| 4. Current consumption - transmitting | 14 |
| 5. Current consumption - receiving | 14 |
| 6. Current consumption - low power | 14 |
| 7. RSSI accuracy | 16 |
| 8. Timing | 16 |
| 9. Transmit and receive power | 16 |
| 10. Sensitivity at different data rates | 16 |
| 11. Pin characteristics | 17 |
| 12. Pinout | 19 |
| 13. Radio profiles | 28 |
| 14. Security levels | 31 |
| 15. UART parameters | 34 |
| 16. Confirmation status byte | 37 |
| 17. Requests | 73 |
| 18. Confirmations | 74 |
| 19. Indications | 75 |
| 21. Channel index and Frequency of operation | 86 |
| 22. ETX configuration flags | 103 |
| 23. Table of settings | 110 |
| 24. Supported GPIOs | 117 |
| 25. Start up timing | 123 |
| 26. Maximum throughput timings, packet error rate = 0% | 125 |
| 27. Maximum throughput timings, packet error rate = 0% | 126 |
| 28. Classification reflow soldering profile, Note: refer to IPC/JEDEC J-STD-020E | 146 |
| 29. Dimensions | 150 |
| 30. Weight | 150 |
| 31. Lot number details | 153 |
| 32. CRC8 Test Vectors | 169 |

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 32: 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