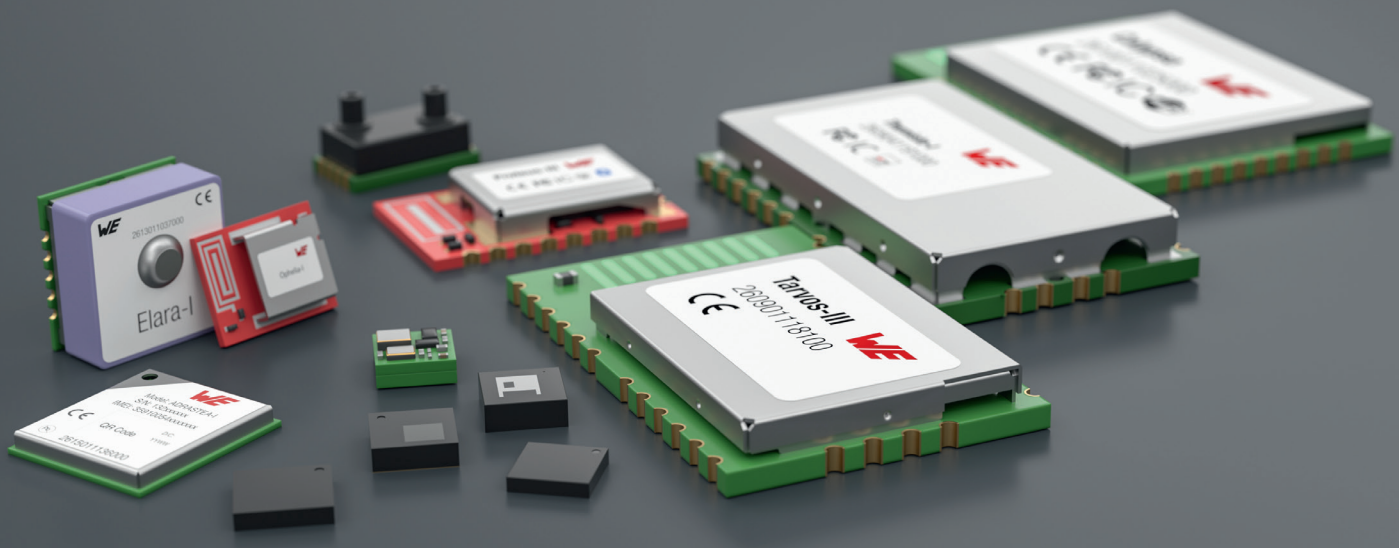


User manual

Cordelia-I

2610011025010

Version 1.2

January 30, 2026

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT

****

MUST READ

Check for firmware updates

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

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

Contents

Overview of helpful application notes

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.

Ground plane effects on radio module antennas

http://www.we-online.com/ANR033
The ground plane plays a critical role in the performance of radio module antennas, affecting parameters such as radiation pattern, gain, and efficiency. This application note provides practical insights into how ground plane size, shape, and placement influence antenna behavior, offering guidance for optimal integration in real-world designs. Simulation results and measurement data are included to illustrate key effects and support design decisions.

1 Revision history

Manual version	FW version	HW version	Notes	Date
1.0	1.0.0	2.2	• Initial release of the manual	November 2024
1.1	1.1.0	2.2	• Updated `EU Declaration of conformity` • Added information related to the EN18031-1	July 2025
1.2	1.1.0	2.2	• fixed missing pinout picture in chapter 3	January 2026

2 Abbreviations

Abbreviation	Name	Description
0xhh [HEX]	Hexadecimal	All numbers beginning with 0x are stated as hexadecimal numbers. All other numbers are decimal
AP	Access Point	WLAN (IEEE 802.11) infrastructure node offering stations to connect to
BDM	Business development manager	Support and sales contact person responsible for limited sales area
CA	Certification authority	
DC	Duty cycle	Transmission time in relation of one hour. 1 % means, channel is occupied for 36 s per hour
DHCP	Dynamic host configuration protocol	Application layer protocol
DNS	Domain name system	Application layer protocol
FOTA	Firmware over the air	Update mechanism
HIGH	High signal level	Digital voltage level that is detected as high by the module
HTTP(s)	Hypertext transfer protocol (secure)	Application layer protocol
IEEE	Institute of electrical and electronics engineers	
IP	Internet protocol	Network layer protocol
LOW	Low signal level	Digital voltage level that is detected as low by the module
LSB	Least significant bit	
MAC	Medium access control	
MQTT	Message queuing telemetry transport	Application layer protocol
MSB	Most significant bit	
NWP	Network processor unit	802.11 network processor unit
PKI	Public key infrastructure	Authentication mechanism
REST	Representational state transfer	
RF	Radio frequency	Describes everything relating to the wireless transmission
STA	Station	WLAN (802.11) node in station role, can connect to an AP
SaaS	Software as as service	

TLS	Transport layer security	Transport layer protocol
UART	Universal asynchronous receiver transmitter	Serial interface protocol
US	User settings	Any relation to a specific entry in the user settings is marked in a special font and can be found in the respective chapter
VDD	Voltage Drain Drain	Supply voltage
WEP	Wired Equivalent Privacy	802.11 security algorithm
Wi-Fi		Is a Registered Trademark of the Wi-Fi Alliance for interoperability tested WLAN (IEEE 802.11) based products
WLAN	Wireless Local Area Network	
WPA	WiFi Protected Access	WiFi security algorithm
WPS	WiFi Protected Set-up	WiFi security algorithm

3 Introduction

The Cordelia-I is a compact WLAN radio module based on IEEE 802.11 b/g/n intended to be used as an IoT connectivity sub-module to provide secure cloud connectivity to an embedded system. The module acts as a bridge between a host MCU on the end system and the cloud platform.

The edge castellated connections, smart antenna configuration and an easy-to-use AT-style command interface enables easy integration of Cordelia-I into any embedded application. Whether a serial cable replacement or low power IoT application with cloud connectivity, the Cordelia-I WLAN module offers a robust and standard compliant wireless connectivity solution for low-power and low-medium throughput applications.

The Cordelia-I module is fully compliant to RED including the delegated regulation for cyber-security 2022/30 as per EN18031-1 norm making it a secure basis to build any embedded IoT application.

WLAN will be used as a synonym for IEEE 802.11 standard compliant radio communication throughout this manual.

3.1 Operational description

The Cordelia-I WLAN module is intended to be used as a radio sub-system in order to provide secure cloud connectivity capabilities to the system.

The UART acts as the primary interface between the module and the host MCU. The module can be fully configured and controlled using a set of AT commands over UART.

Once configured, the module independently manages secure connectivity to the cloud allowing the host MCU to utilize its resources elsewhere. Therefore, when using the standard firmware, a host MCU is required in the end-product to control and access the radio module. Standalone applications, without host, can be realized with a custom firmware development.

MQTT over TLS is used as primary protocol for connectivity to the cloud. The module comes with a secure root-of-trust which is unique and tamper proof. These features can be leveraged to create a secure channel to the cloud that data agnostic.

Additionally, the Cordelia-I supports connectivity to the *QuarkLink*^{TM} platform from *Crypto Quantique* out-of-the-box. This ensures secure and scalable zero touch provisioning as well as cloud onboarding of the end-device in the field. For more details refer to chapter 6.

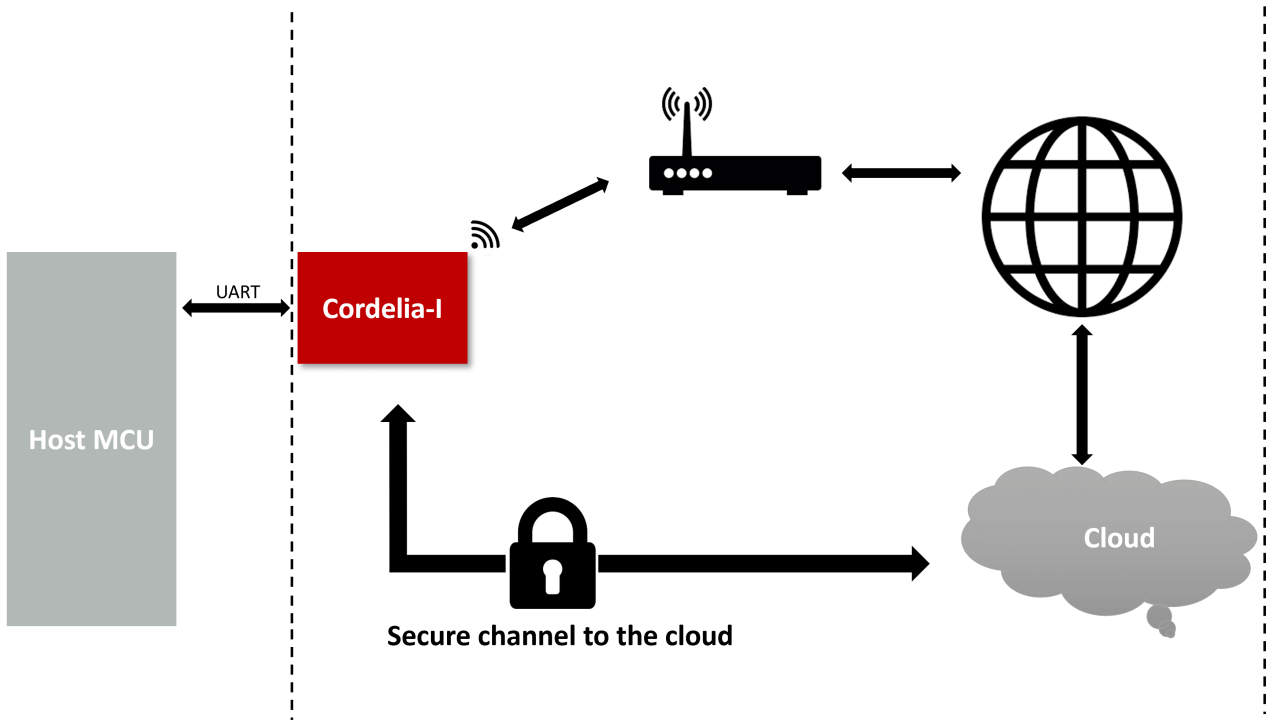


Figure 1: The Cordelia-I IoT module is capable of MQTT data transfer on both encrypted and non-encrypted channels

3.2 Block diagram

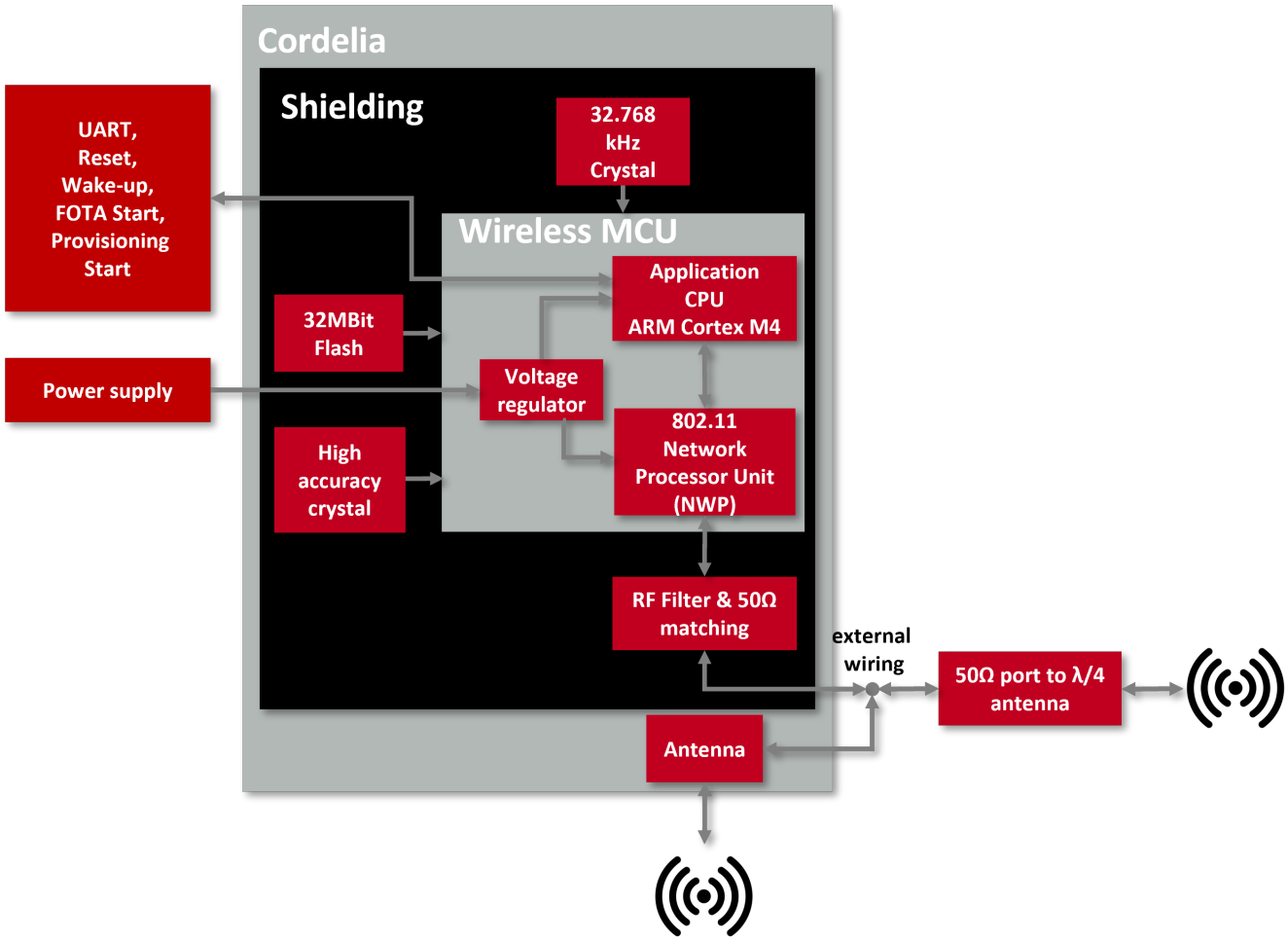


Figure 2: Block diagram

3.3 Ordering information

WE order code	Description
2610011025010	WLAN module in T&R packaging
2610019225011	EV-Kit for WLAN module

Table 3: Ordering information

4 Electrical specifications

Unless otherwise stated, all the values given here were measured on the Cordelia-I EV-Board under the following conditions: T = 25 ℃, VDDS= 3.6 V, internal DC-DC converter active and 50 Ω conducted.

4.1 Operating conditions

Description	Min.	Typ.	Max.	Unit
VCC	2.1	3.3	3.6	V
Temperature range	-40	25	85	℃
Ambient thermal slew rate	-20		20	℃ / min

Table 4: Operating conditions

When operating at an ambient temperature of over 75 ℃, the transmit duty cycle must remain below 50 % to avoid enabling the auto-protect feature of the power amplifier. If the auto-protect feature is triggered, the device takes a maximum of 60 s to restart the transmission.

The VCC brown-out threshold is 2.1V, the VCC blackout level is 1.67V. As ripples may apply when dips happen (e.g. when the TX state is entered) the current supply shall ensure at least VCC of 2.1V is always present in any operating state of the radio module.
To ensure the module's radio performance, ripple on the supply must be less than ± 300 mV.

4.2 Absolute maximum ratings

Description	Min.	Typ.	Max.	Unit
VCC	-0.5		3.8	V
Digital inputs	-0.5		VCC	V
Pin RF, Pin ANT	-0.5		2.1	V

Table 5: Absolute maximum ratings

4.3 Power consumption

4.3.1 Static

Description	Min	Typ.	Max	Unit
TX current consumption at max output power (1 Mbit DSSS mode)		230		mA
RX current consumption (1 Mbit DSSS mode)		76		mA
Sleep mode (WLAN disconnected)		10		µA
Power save mode (WLAN station connected, socket connected, UART off)		2		mA
Peak calibration current, *VCC*=2.1V		670		mA
Peak calibration current, *VCC*=3.3 V		450		mA
Peak calibration current, *VCC*=3.6 V		420		mA

Table 6: Power consumption

4.4 Radio characteristics

Description	Min	Typ.	Max	Unit
Max output power		16	18	dBm
Input sensitivity (1 Mbit)	-94	-92		dBm
Max input level, 802.11b		-4		dBm
Max input level, 802.11g		-10		dBm
Frequencies	2412		2472	MHz

Table 7: Radio characteristics

Standard	Modulation and coding	Peak Data rate
802.11b	DBPSK(DSSS)	1 Mbps
	DQPSK(DSSS)	2 Mbps
	DQPSK(CCK)	5.5 Mbps
	DQPSK(CCK)	11 Mbps
802.11g	BPSK(OFDM) coding rate 1/2	6 Mbps
	BPSK(OFDM) coding rate 3/4	9 Mbps
	QPSK(OFDM) coding rate 1/2	12 Mbps
	QPSK(OFDM) coding rate 3/4	18 Mbps
	16-QAM(OFDM) coding rate 1/2	24 Mbps
	16-QAM(OFDM) coding rate 3/4	36 Mbps
	64-QAM(OFDM) coding rate 2/3	48 Mbps
	64-QAM(OFDM) coding rate 3/4	54 Mbps
802.11n	BPSK(OFDM) coding rate 1/2	7.2 Mbps
	QPSK(OFDM) coding rate 1/2	14.4 Mbps
	QPSK(OFDM) coding rate 3/4	21.7 Mbps
	16-QAM(OFDM) coding rate 1/2	28.9 Mbps
	16-QAM(OFDM) coding rate 3/4	43.3 Mbps
	64-QAM(OFDM) coding rate 2/3	57.8 Mbps
	64-QAM(OFDM) coding rate 3/4	65 Mbps
	64-QAM(OFDM) coding rate 5/6	72.2 Mbps

Table 8: Modulation schemes and peak data rate.

4.5 Pin characteristics

Property	Min	Typ.	Max	Unit
RF, ANT pin input voltage			2.1	V
GPIO voltage input high	0.65 × *VCC*		*VCC*	V
GPIO voltage input low	-0.5		0.35 × *VCC*	V
/RESET voltage input high		*VCC*		V
/RESET voltage input low		0.6		V
GPIO voltage output high	0.8 × *VCC*		VCC	V
GPIO voltage output low	0		0.2 × *VCC*	V
Pin output current sunk by any I/O and control pin, drive mode dependent		2		mA
Pin output current sourced by any I/O and control pin, drive mode dependent		2		mA

Table 9: Pin characteristics, VDDS = 3.3 V, T = 25 ℃

4.6 TX power vs current consumption

The following tables contains the typical TX power values and the corresponding typical average current for 3.6 V supply voltage and 25 ℃ ambient temperature. Cable losses of the conducted measurement are about 2 dB.

Tx power index	TX power [dBm]	Average current [mA]
0	13.97	260.15
1	12.59	255.95
2	11.62	249.5
3	11.53	251.17
4	10.57	189.35
5	9.47	184.4
6	8.93	182.3
7	8.96	182.3
8	8.89	182.27
9	8.88	182.22
10	8.81	182.29
11	8.86	182.2
12	8.88	182.17
13	8.89	182.18
14	8.92	182.2
15	8.93	182.11

Table 10: TX power vs current consumption, conducted measurement of continuous data transmission, rate 1Mbps (DSSS)

Tx power index	TX power [dBm]	Average current [mA]
0	11.74	119.74
1	10.48	118.95
2	9.46	118.36
3	8.36	117.91
4	8.87	103.10
5	8	102.29
6	6.80	101.73
7	5.83	101.29
8	4.93	100.84
9	3.93	100.59
10	2.88	100.30
11	1.98	100.18
12	1.09	100.02
13	0.75	100
14	0.73	100
15	0.64	100

Table 11: TX power vs current consumption, conducted measurement of continuous data transmission, rate 54 Mbps (OFDM)

5 Pinout

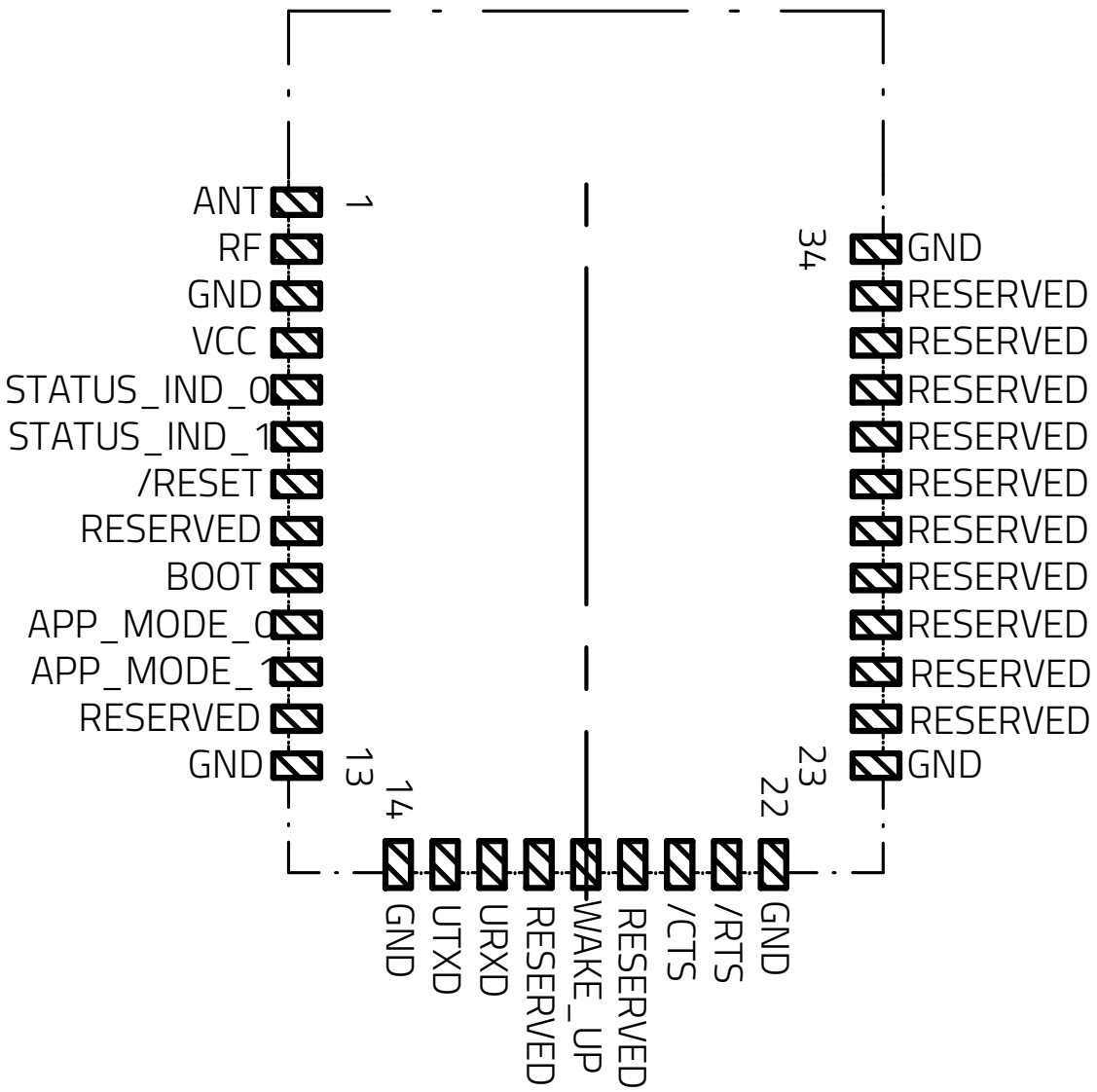


Figure 3: Pinout (top view)

Pad	Name	Chip-set pin	Description
1	*ANT*	-	RF connection to PCB antenna (see section 7.1)
2	*RF*	-	50 Ω RF connection to external antenna or on-board antenna via ANT (see section 7.1)
3	*GND*	-	Negative supply voltage
4	*VCC*	-	Positive supply voltage
5	*STATUS_IND_0*	GPIO8	Status indication LED 0, do not connect if not needed
6	*STATUS_IND_1*	GPIO9	Status indication LED 1, do not connect if not needed
7	*/RESET*	-	Reset (active low), internal pull-up (100 kΩ)
8	*WAKE_UP*	GPIO12	Unused, output LOW, do not connect if not needed
9	*BOOT*	-	Input with internal pull-down (2.7 kΩ), pull low during start-up to boot the standard application, do not connect if not needed
10	*APP_MODE_0*	GPIO22	Input, internal weak pull-down (see section 6.4), do not connect if not needed
11	*APP_MODE_1*	GPIO0	Input, internal weak pull-down (see section 6.4), do not connect if not needed
12	*RESERVED*	GPIO30	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
13	*GND*	-	Negative supply voltage
14	*GND*	-	Negative supply voltage
15	*UTXD*	GPIO2	Module UART TX, set to HIGH in case Cordelia-I UART is disabled
16	*URXD*	GPIO1	Module UART RX, uses internal weak pull-up
17	*RESERVED*	GPIO3	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
18	*WAKE_UP*	GPIO4	Wake-up on rising edge, internal pull-down, do not connect if not needed
19	*RESERVED*	GPIO5	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
20	*/CTS*	GPIO6	Optionally UART CTS (see section 9.2), uses internal weak pull-down, do not connect if not needed

21	*/RTS*	GPIO7	Optionally UART RTS (see section 9.2), do not connect if not needed, set to HIGH in case Cordelia-I UART is disabled
22	*GND*	-	Negative supply voltage
23	*GND*	-	Negative supply voltage
24	*WAKE_UP*	GPIO10	Unused, output LOW, do not connect if not needed
25	*WAKE_UP*	GPIO11	Unused, output LOW, do not connect if not needed
26	*RESERVED*	GPIO14	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
27	*RESERVED*	GPIO15	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
28	*RESERVED*	GPIO16	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
29	*RESERVED*	GPIO17	Unused, output LOW, do not connect if not needed (with customized firmware, usage as GPIO is possible)
30	*RESERVED*	JTAG_TDI	Debug line (locked), do not connect
31	*RESERVED*	JTAG_TDO	Debug line (locked), do not connect
32	*RESERVED*	JTAG_TCK	Debug line (locked), internal pull-down, do not connect
33	*RESERVED*	JTAG_TMS	Debug line (locked), do not connect
34	*GND*	-	Negative supply voltage

Table 12: Pinout

6 Functional description

The Cordelia-I WLAN module is intended to be used as a radio sub-system in order to provide secure cloud connectivity capabilities to the system.

6.1 Key features

In this section, the features of the Cordelia-I module are summarized in the form of a table. Cordelia-I allows the user to configure and exploit its rich features through an easy-to-use command interface over UART.

Feature	Description
Radio standards	IEEE 802.11 b/g/n station
Radio Channels	1-13
Security	WEP, WPA/WPA2PSK, WPA2 Enterprise (802.1x), WPA3
Provisioning	In AP mode using the on-board HTTP server
Network layer	IPv4, IPv6
IP addressing	Static, LLA, DHCPv4, DHCPv6 with DAD
Network applications	MQTT(S) after zero-touch provisioning using QuarkLink
Software Update	Secure FOTA update with fall back mechanism in AP as well as Station modes
Power management	802.11 power save modes Sleep mode with timed or pin wake-up
Transparent mode	Transparent UART to cloud mode for cable replacement applications
Security	Secure key storage Trusted root-certificate catalog Encrypted file system Secure OTA Software tamper detection Cloning protection
Secure cloud connectivity	Zero touch device provisioning and secure cloud on boarding using the QuarkLink™ platform

Table 13: Key features

6.2 Functional interfaces

The Cordelia-I module has the following functional interfaces.

Interface	Factory default state
AT command interface over UART	Enabled
Transparent UART interface	Disabled
WLAN interface	Disabled
MQTT interface for cloud connectivity	Disabled
HTTPS interface for FOTA	Disabled
HTTPS interface to QuarkLink™	Disabled

Table 14: Functional interfaces

In order to provide secure cloud connectivity, the module needs to undergo a one-time configuration. In the following sections, a detailed set of the parameters available for each of these interfaces can be found.

6.2.1 AT command interface

The UART interface has the following parameters.

- **baudrate**: is set to 115200 bit/s by default can take a value up to 3 Mbit/s. Use of flow control is recommended for rates higher than 921600 bit/s.
- **parity**: is set to "none" by default.
- **flowcontrol**: is not set by default.

6.2.2 Transparent UART interface

The following parameters need to be configured to use the transparent UART interface.

- **baudrate**: is set to 115200 bit/s by default can take a value up to 3 Mbit/s. Use of flow control is recommended for rates higher than 921600 bit/s.
- **parity**: is set to "none" by default.
- **flowcontrol**: is not set by default.
- **transparent_trigger**: is a flag that determines the trigger to start wireless data transmission in the transparent mode. The trigger can be receipt one or two specific end-of-transmission characters or a timeout. The default value us "2etx|timer" (trigger on 2 end-of-transmission characters and timeout)
- **transparent_timeout**: is the time after which a transmission is triggered. Default value is 20 ms.
- **transparet_etx**: The set of characters that trigger a transmission in the transparent mode. The default value is 0x0D0A ("\r\n")

6.2.3 MQTT interface

Parameters of the MQTT interface are as described below.

- **iotHubEndPoint**: is the address of the MQTT broker. This field is mandatory parameter for all connections and is a string of maximum length 512 byte.
- **iotHubPort**:is the port of the MQTT broker. This is a mandatory parameter for all connections and set to 8883 by default. It can take a value of range 0 - 65535.
- **rootCAPath**: is the path to the rootCA certificate (mandatory parameter for secure connections). The maximum length is 180 byte. Make sure that the root CA certificate is uploaded to this path.
- **clientCertPath**: is the path to the client certificate (mandatory parameter for secure connections). The maximum length is 180 byte. Make sure that the private key is uploaded to this path.
- **clientPrivateKey**: is the path to the client key (mandatory parameter for secure connections). The maximum length is 180 byte. Make sure that the client certificate is uploaded to this path.
- **clientID**: is the client ID string used for MQTT connection (mandatory parameter for all connections). The maximum length is 256 byte.
- **flags**: are a combination of one or more flags (mandatory parameter for all connections). For example, for a secure connection to a server with address "test.mqtt.server" use the flag "url|sec". The | inbetween two paramters represents an "AND" concatonation.
  - **url**: Use this flag when the MQTT broker address is a URL.
  - **ip4**: Use this flag when the MQTT broker address is an IPv4 address.
  - **ip6**: Use this flag when the MQTT broker address is an IPv6 address.
  - **sec**: Use this flag when the connection needs to be secure (TLS 1.2).
  - **skip_domain_verfiy**: Use this flag to skip verification of the domain name.
  - **whitelist_rootca**: Use this flag to whitelist the provided root CA bypassing the root CA catalog verification.
  - **skip_date_verfiy**: Use this flag to skip date verification.
- **base64**: can be a 0 = Payload is binary or 1 = payload is base64 encoded (default value is 0).
- **willTopic**: is the topic for the will message (this setting is optional). The maximum length is 512 byte.
- **willMessage**: is the payload for the will message (this setting is optional). The maximum length is 512 byte.
- **willQoS**: can be 0, 1 or 2 corresponding to QoS0, QoS1 or QoS2 (this setting is optional)

- **userName**: is the username for the MQTT connection (this setting is optional). The maximum length is 512 byte.
- **pasword**: is the password for the MQTT connection (this setting is optional). The maximum length is 512 byte.
- **keepAliveTimeout**: is the keep alive timeout. This can take a value of the range 0 - 65535.
- **cleanConnect**: determines if the session requires a clean connect (0=false, 1=true). The default value is 1.
- **subtopic name**: name of the topic to subscribe. The maximum length is 512 byte.
- **subtopic QoS**: can be 0, 1 or 2 corresponding to QoS0, QoS1 or QoS2.
- **pubtopic name**: name of the topic to publish. The maximum length is 512 byte.
- **pubtopic QoS**: can be 0, 1 or 2 corresponding to QoS0, QoS1 or QoS2.
- **pubtopic retain**: can be 0, 1 to enable or disable message retention at the broker.

The module supports configuration of up to 4 different topics to subscribe or publish. In case the parameter "flags" parmeter contains "sec" for a secure connection, additional parameters for the SNTP server and certificate must be specified before a secure socket can be setup.

6.2.4 QuarkLink parameters

The connection to QuarkLink™ platform requires the configuration of the following parameters.

- **hostname**: is the URL of the QuarkLink instance. For example, "example.QuarkLink.io". The maximum length is 512 byte.
- **iotHubPort**: is the port of the QuarkLink instance. This is a mandatory parameter for all connections and set to 6000 by default. It can take a value of range 0 - 65535.
- **rootCAPath**: is the path to the rootCA certificate (mandatory parameter for secure connections). The maximum length is 180 byte. Make sure that the root CA certificate is uploaded to this path.

Further, to perform the enrolment process, the following parameters needs to be set. These parameters will be a part of the device certificate that will be issued by the QuarkLink™ platform.

- **country**: A two/three digit country code. For example, for Germany, the value of country code is "DE".
- **state**: This is the string that contains the state name that can be a maximum 128 byte long.
- **locality**: This is the string that contains the state that can be a maximum 128 byte long.

- **surname**: This is the string that contains the surname that can be a maximum 64 byte long.
- **email**: This is the string that contains the email that can be a maximum 256 byte long.
- **organization**: This is the string that contains the organization that can be a maximum 64 byte long.
- **unit**: This is the string that contains the unit that can be a maximum 64 byte long.

6.2.5 FOTA parameters

In order to enable the module to perform FOTA updates, the following paramters need to be set.

- **url**: is the URL of the FOTA server. This value by default is https://www.we-online.com/res/wco/Cordelia
- **rootCAPath**: is the path to the rootCA certificate (mandatory parameter for secure connections). The maximum length is 180 byte. The root CA certificate corresponding to the default FOTA server is present on the module by default.

6.3 Modes of configuration

The Cordelia-I module can be fully configured using one of the following methods.

1. **AT commands:** Using this method, the module can be fully configured and controlled from a host MCU. More details on this interface can be found in chapter 10.
2. **Local Provisioning:** The Cordelia-I module can be booted up in local provisioning mode in which it appears as an access point. Any WLAN enabled device can be used to connect to this A.P and configure the device by accessing the on-board HTTP server from a browser. See chapter 12 for more details.
3. **Remote Provisioning using QuarkLink™:** The MQTT interface of the module can be remotely configured using the QuarkLink™ platform.Please note the module needs to be connected to the internet via WLAN network to access this feature. However, the QuarkLink™ platform offers provisioning of WLAN and other interfaces via UART when connected via the browser. More details on this in the chapter 6.7.

It is recommended to use the AT command interface to initially configure the device and the QuarkLink™ platform to remotely configure the MQTT interface. This method ensure maximum security, scalability as well as flexibility.

6.4 Modes of operation

When active, the Cordelia-I can be in one of the following operation modes. The transition to/from the modes occurs due to one of the following reasons.

- Level of the *APP_MODE_x* pins during boot up
- */Reset* signal
- *WAKE_UP* signal or time event

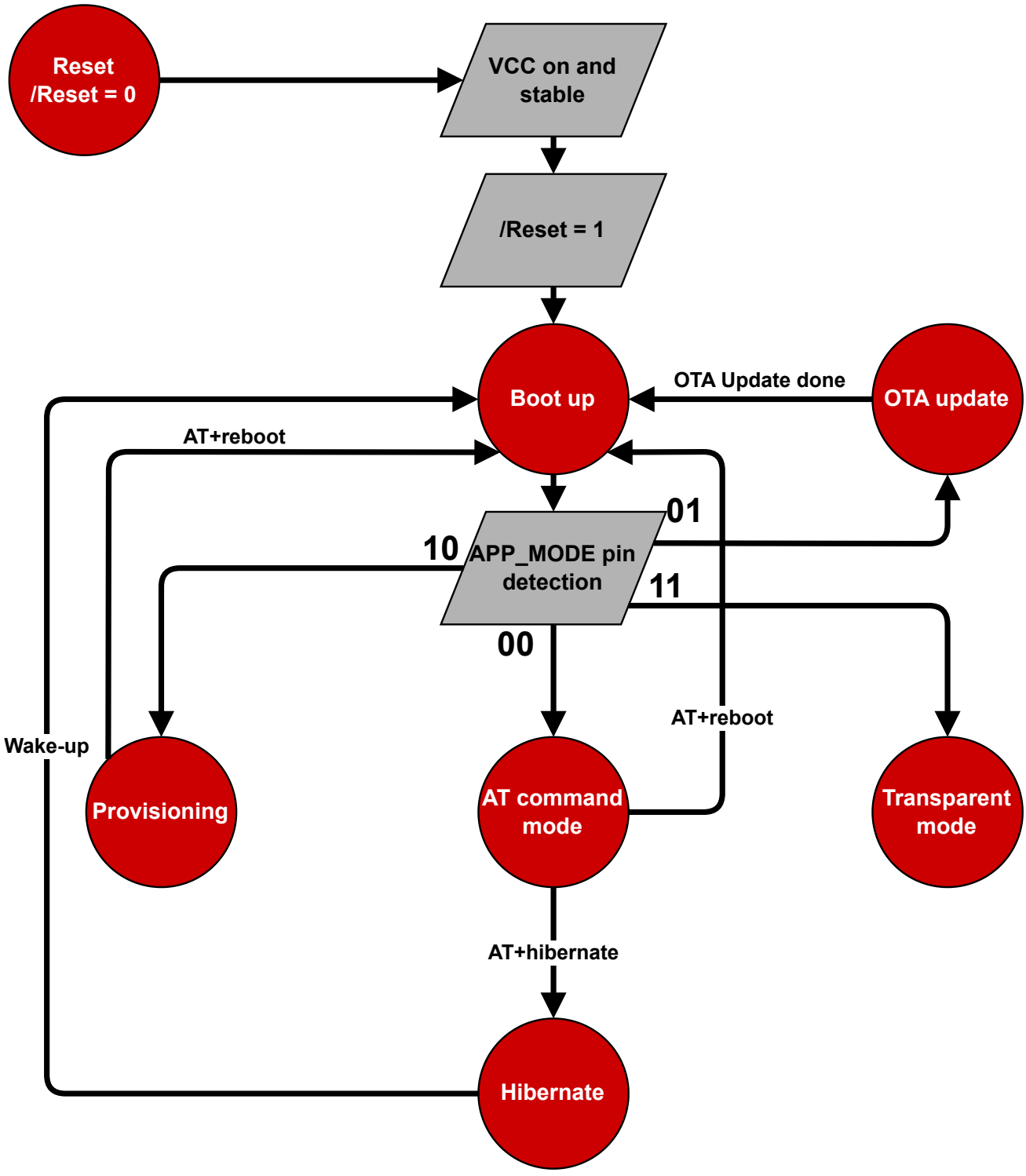


Figure 4: Modes of operation

6.4.1 BootUp

Based on the voltage level on the pins *APP_MODE_0* and *APP_MODE_1*, the module boots up in one of the following modes.

Mode index	*APP_MODE_1*	*APP_MODE_0*	Description
0	LOW	LOW	AT command normal mode, see chapter 10
1	LOW	HIGH	FOTA mode, see chapter 16
2	HIGH	LOW	Provisioning mode, see chapter 12
3	HIGH	HIGH	Transparent mode, see chapter 13

Table 15: Application modes

6.4.2 AT command mode

In this mode, Cordelia-I allows the user to configure and control the module using a AT based command interface over UART. The AT-command interface is described in detail in chapter 10. A transition to hibernate can be done using the appropriate commands.

6.4.3 FOTA update mode

In this mode of operation, the module allows secure over the air firmware update by downloading an image stored in a remote server. Further details regarding the OTA update mechanism can be found in chapter 16.

6.4.4 Provisioning mode

To enable easy provisioning when integrated into an embedded system with limited HMI capabilities, the Cordelia-I offers a provisioning mode. In this mode, the module acts as an AP and allows external devices with appropriate credentials to connect and access the on-board HTTP server. The user can conveniently browse the settings web page and configure the module using any web browser. More details regarding provisioning can be found in chapter 12.

6.4.5 Hibernate mode

It is essential to have a low power sleep mode for battery powered systems. Cordelia-I offers a hibernate mode with a very low current consumption of less than 10 µA. The characteristics of this mode are as described below.

- The network processor is in hibernate mode and the application processor is shut down.
- The module wakes up automatically after a time period specified in the hibernate command.
- Alternatively, the module can be woken up manually with a rising edge on the *WAKE_UP* pin.
- UART RX and TX pin needs to be properly terminated to prevent any leakage current.
  - *UTXD* - HIGH
  - *URXD* - HIGH
- The *RTS* pin has the following state:

  - UART configured without flow control *UTXD* - LOW
  - UART configured with flow control *URXD* - HIGH

- On wake up, the module starts from the reset vector meaning that the RAM contents are lost.
- Based on the WiFi connection policy (see chapter 11.3.5), the module can be set up to automatically connect to a saved access point profile and acquire an IP address.
- The cloud connections are lost on entering sleep mode and have to be re-established on wake-up.

Section 11.1.5 describes the commands used to put the module to hibernate and chapter 15 describes the timing characteristics.

6.4.6 Transparent mode

In transparent mode, the Cordelia-I automatically connects to a preconfigured access point and opens a secure cloud connection for communication with a preconfigured remote MQTT endpoint. Afterwards, the Cordelia-I acts as a transparent bridge between the UART and the created socket. This means that all data sent to the Cordelia-I via UART is forwarded to the socket and all data received on the socket is output on the UART. For more details see chapter 13.

To use transpareent mode the Cordelia-I must have been configured properly before by one of the available methods. This interface will not provide any feedback to the user.

6.5 Secure cloud connectivity

In order to make the connection to the cloud secure, the device and cloud should perform mutual authentication followed by exchange of a session key which is then used to encrypt the communication channel. This is typically performed using the TLS protocol.

In order to establish a mutual TLS connection, the following cryptographic assets are required to be present on the device and the cloud respectively. The assets also need to be matching with each other.

Cryptographic assets on the device.

- **Device ID:** Unique device identity that is immutable and tamper-proof.
- **Device key:** This is a private key that is unique to every device and needs to be secret.
- **Device certificate:** This certificate contains the public key corresponding to the device private key.
- **Root CA:** This is the root CA of the cloud used to authenticate the cloud end point.

The process of storing these parameters on to the end-device is called **device provisioning**.

Cryptographic assets on the cloud end point.

- **List of Device IDs:** A white list containing Device IDs to allow for connections.
- **Device certificates:** Public keys corresponding to the device IDs to enable device authentication.

The process of storing these parameters on to the cloud end point is called **cloud onboarding**. These cryptographic assets need to be securely stored on to the device as well as the cloud end point. The exposure of these assets in any phase of the device manufacturing life-cycle can potentially break the security. Often, human interaction with these cryptographic assets poses the largest threat. Hence, the following steps needs to be followed to ensure maximum security.

The Cordelia-I module with the QuarkLink™ platform enable secure cloud connectivity using Zero touch device provisioning and secure cloud onboarding. Further, a complete device management including remote cloud migration can be performed during the entire life-cycle of a Cordelia-I.

6.5.1 Zero touch device provisioning

This process involves configuring the device with all the parameters required to connect to the cloud platform including the cryptographic assets without any human interaction.
This involves configuring several parameters including the following cryptographic assets on the device.

- **Device ID:** Unique device identity that is immutable and tamper-proof.
- **Device key:** This is a private key that is unique to every device and needs to be secret.
- **Device certificate:** This certificate contains the public key corresponding to the device private key.
- **Root CA:** This is the root CA of the cloud used to authenticate the cloud end point.

The Cordelia-I module along with QuarkLink™ platform enable zero touch provisioning in the field. Each module comes with a unique pre-installed and hardware tamper-proof set of keys. The private key is fixed to the hardware and cannot be read out by the application. The application software can access only the public key and can use the private key for further cryptographic operations.

Device ID: is generated by performing a one-way mathematical function on the device public key and stored in the secure file system of the Cordelia-I module. The command to read the device unique ID can be found in section 11.2.1.

Device key: is the private key that is pre-installed into the hardware and made tamper-proof. This cannot be read out even by the module application but leveraged to perform device authentication during secure cloud connectivity.

Device certificate: is created by the QuarkLink™ platform during the enrolment process described in section 6.7.

Root CA: is loaded by the QuarkLink™ platform during the enrolment process described in 6.7.

6.5.2 Secure cloud onboarding

This process involves configuring the cloud end point to allow authenticated connection on end-devices. This process involves storing the following parameters in the cloud end point.

- **List of Device IDs:** A white list of devices to allow connection.
- **Device certificates:** Public keys corresponding to the device IDs to enable device authentication.

The QuarkLink™ platform allows secure device onboarding on to most of the popular cloud service provider including self-hosted servers. This is handled using authenticated REST APIs exposed by the cloud service providers. More information on adding a cloud end point can be found in the QuarkLink™ documentation under *https://docs.QuarkLink.io/docs/iothubs*.

6.6 QuarkLink secure connectivity platform

QuarkLink™ is a SaaS secure connectivity platform for IoT devices. It enables companies and enterprises to seamlessly perform the first layer of security for an IoT device to be secure when being deployed into a supply chain. These steps include securely provisioning devices, onboarding those devices to a cloud service provider or their own server-hosted application, then managing those devices over their lifecycle. QuarkLink uses advanced cryptography techniques to integrate with any root-of-trust for zero-trust end-to-end security across every IoT device.

6.7 Remote provisioning using the QuarkLink platform

The QuarkLink™ in combination with Cordelia-I module, enables remote zero touch provisioning as well as secure on-boarding of the device. The so called "enrolment" process is used to perform these actions.

6.7.1 Device enrolment process

The enrolment process automates the following tasks and performs the same in a secure manner.

- Perform zero touch provisioning to load all the necessary cryptographic assets to the module (see section 6.5.1).
- Fully configure the MQTT interface remotely by setting all the necessary parameters (see section 6.2.3).
- Perform secure cloud onboarding as described in section 6.5.2.

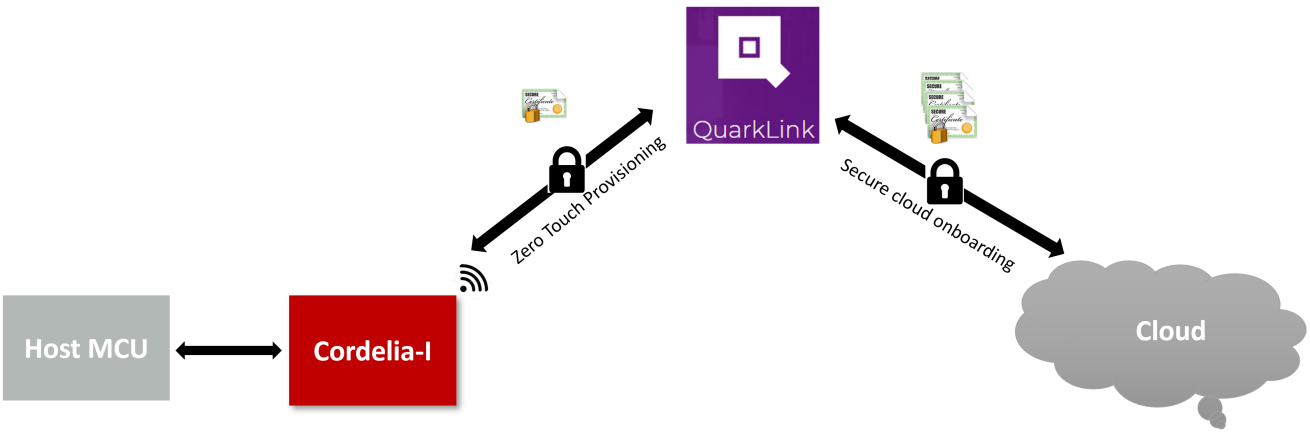


Figure 5: Enrolment

This process can be triggered from the host MCU using a single AT command (refer to section 11.7).

6.8 Secure data exchange

Once the module is configured either manually or using the enrolment process, it is ready to open a secure communication channel to the cloud end point. This can be done using the AT commands(Refer to section 11.6). On successful connection, data exchange can begin over the secure channel.

Alternatively, the module can be switched to the transparent mode where, it creates the secure connection automatically allowing user data exchange via UART.

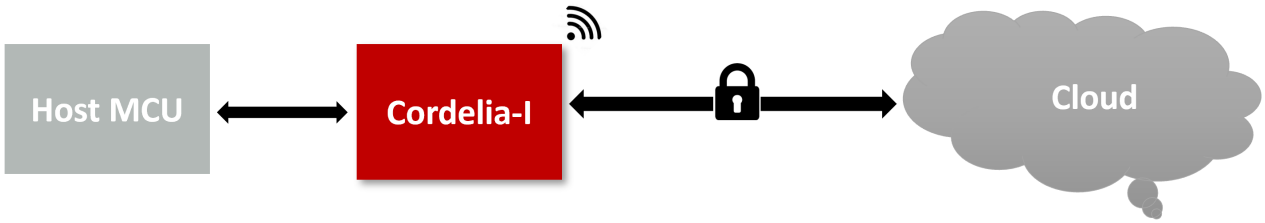


Figure 6: Secure cloud channel for data exchange

7 Quick start guide

The Cordelia-I WLAN module comes pre-flashed, tested and ready-to-use out-of-the-box. This chapter describes steps to quickly build a prototype system and test the capabilities of the module.

7.1 Antenna connection

Cordelia-I's smart antenna configuration enables the user to choose between two antenna options:

7.1.1 On-board PCB antenna

The Cordelia-I has an on-board PCB antenna optimized for operation in the 2.4 GHz band. A simple short between the pins *RF* and *ANT* feeds the RF output of the module to the on-board antenna. In this configuration the module does not require any additional RF circuitry.

7.1.2 External antenna

For applications that use an external antenna, the Cordelia-I provides a 50 Ω RF signal on pin *RF*. 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 21 for further information.

7.2 Minimal pin configuration

The following pins must be connected as described in table 16 for correct operation. The remaining can be left unconnected.

Pin number	Pin function	Pin connection
1	*ANT*	Connect to pin 2 or *GND* (see chapter 7.1)
2	*RF*	Connect to pin 1 or external antenna (see chapter 7.1)
3,13,14,22,23,34	*GND*	*GND*
4	*VCC*	*VCC*
7	*/RESET*	Host GPIO and/or reset button
5,6	*STATUS_IND_x*	Optionally to host GPIO or LED for status indication
9	*BOOT*	Boot pin to host GPIO or GND
10,11	*APP_MODE_x*	Host GPIO for mode selection
15	*UTXD*	Host UART RX
16	*URXD*	Host UART TX
18	*WAKE_UP*	Host GPIO for wake up trigger

Table 16: Minimal pin configuration

7.3 Power up

Set and hold the */RESET* pin to LOW. After the supply voltage to the module has stabilized, the */RESET* pin shall be held LOW level for another t_{reset} of at least 200 ms to ensure a safe start-up. Before releasing the */RESET* pin, make sure that the appropriate voltage levels are applied on pins *App_Mode_0* and *App_Mode_1* according to the desired application mode (see section 6.4). Also make sure that the host's UART TX line to the module is configured as a logic HIGH level during module boot-up for indicating UART idle towards Cordelia-I.
The module will send a start-up UART message once it has booted and started the application. The radio module is ready to receive AT commands 200 µs (Δt) after the startup message has been transmitted.
For further timing information refer to chapter 15.

If the module is used on a battery-powered system, using a suitable reset-IC (or a discrete RC block for a delay) is highly recommended to ensure a correct power up and stable behavior independent of the battery status.

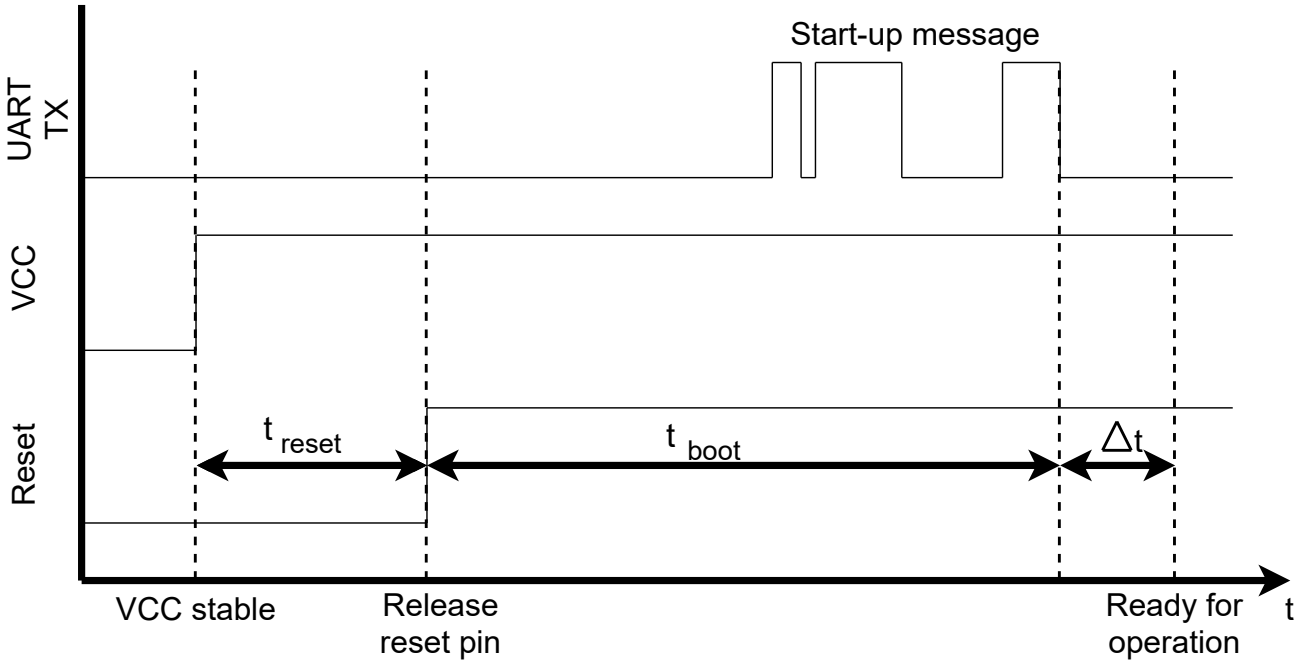


Figure 7: Power up

7.4 Region specific WLAN settings

Despite the world-wide availability of the 2.4 GHz frequency band, there are region specific restrictions on availability of certain channels. In order to be compliant with local regulations, the country code on the module has to be set-up before deployment.

Country code	Supported channels
US	1-11
JP	1-13
EU (Rest of the world)	1-13

Table 17: Country codes

By default, the country code is set to "US" as the channels allowed in this setting is supported world-wide. Country code can be changed by sending the following command to the module. Refer to section 11.3.4 for more details. On request, the modules can be produced with the application-specific country code (see chapter Custom firmware and configuration).

```
AT+wlanSet=general,country_code,EU
OK
```

7.5 Quick start example

This section is intended to demonstrate the zero touch provisioning feature of the Cordelia-I module in combination with the Quarklink platform from CryptoQuantique. Minimal pin and antenna connections have to be done on both the modules as described in sections 7.1 and 7.2. It is recommended to use the Cordelia-I EV-Kit for quick tests.

7.5.1 Prerequisites

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

1. Cordelia-I EV-Board.
2. FTDI drivers are intalled on the used PC.
3. An IEEE 802.11b/g/n compatible access point working in the 2.4 GHz band.
4. Computer with a USB port.
5. A QuarkLink account.

Get a free developer account if the Quarklink platfrom by signing-up using the following link, *https://signup.quarklink.io/*

7.5.2 Hardware configuration

Make sure that the following jumpers are populated in the corresponding positions on the EV-Board. Refer to the Cordelia-I EV-Board specific manual for a complete hardware description.

1. JP3 current bridge is set.
2. Jumpers are set across pins 1-2 (*URXD*), 3-4 (*UTXD*), 9-10 (*STATUS_IND_0*), 11-12 (*STATUS_IND_1*), 13-14 (*WAKE_UP*) and 15-16 (*BOOT*) of the connector JP1.
3. The EV-Board is powered using the barrel connector using external power supply provided in the EV-Kit.
4. The USB interface is connected to the PC using the USB cable.

7.5.3 Step-by-step tutorial

Welcome to this quick-start tutorial! Here, we'll guide you through the setup of a factory-default Cordelia-I module on its evaluation board, so by the end, you'll be able to securely send and receive data with your own MQTT broker. In this tutorial, you'll use only the QuarkLink portal in your web browser - no extra software required.
We'll begin by setting up a personal QuarkLink instance where you can configure and maintain your Cordelia-I modules. From there, we'll walk through provisioning the module with default parameters, connecting it to a secure network, and enrolling it for secure communication. Finally, we'll establish a secure MQTT channel and verify communication by sending and receiving messages between the module and the broker.
By following these steps, you'll gain hands-on experience with setting up a Cordelia-I module, and you will also learn about QuarkLink's powerful tools for secure device management and communication. Let's get started!

It is recommended to use Edge or Chrome browsers when following through this example, as the setup was tested using them. Using other browsers might cause compatibility problems when setting up the serial interface between QuarkLink and the Cordelia-I module.

1. Open up a web browser on your PC and go to `https://signup.quarklink.io/`! Here, you can register your personal QuarkLink instance for free. After registration, you'll have a platform where you can configure and maintain all of your Cordelia-I modules, with useful features like an inbuilt MQTT broker and a serial terminal emulator.
2. After successful registration, you should be able to access your personal QuarkLink instance via the link provided at the end of the registration process. This link will look something like `https://abcdefghij1k2l.quarklink.io/`.
3. Go to your personal QuarkLink instance using that link and log in with the credentials you provided during registration.

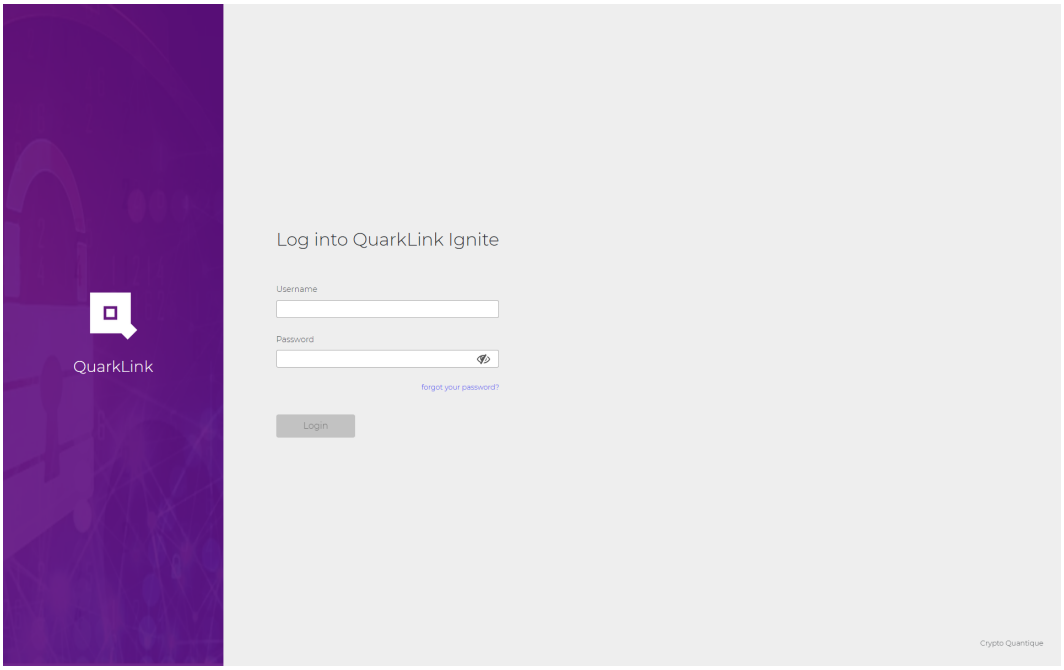


Figure 8: Registering a new QuarkLink instance

4. After logging in, you'll find yourself on the QuarkLink main dashboard. From here, we'll add our Cordelia-I module to the QuarkLink instance and enrol it, activating it for data transfer in your selected IoT Hub (by default, this is the built-in MQTT broker in your QuarkLink instance).

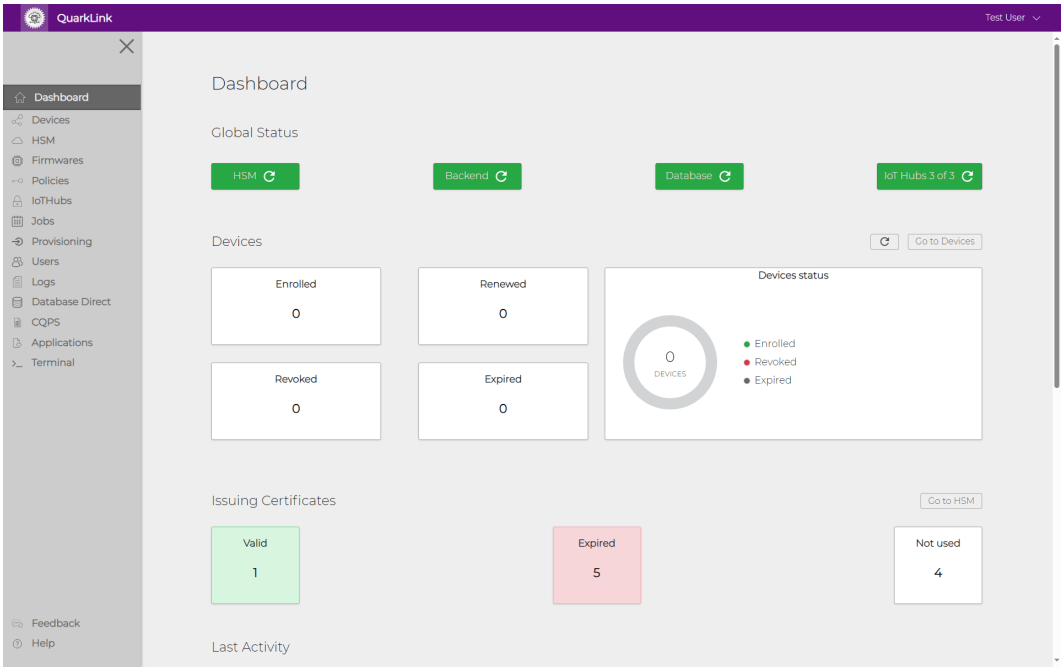


Figure 9: Accessing the QuarkLink main dashboard after login

5. Go to the menu item "Provisioning" and click "New Provisioning Task." Here we'll set up

the default parameters for the Cordelia-I module's configuration. You only need to do this once for each Cordelia-I module; an automated configuration process will run every time you initiate a provisioning task.

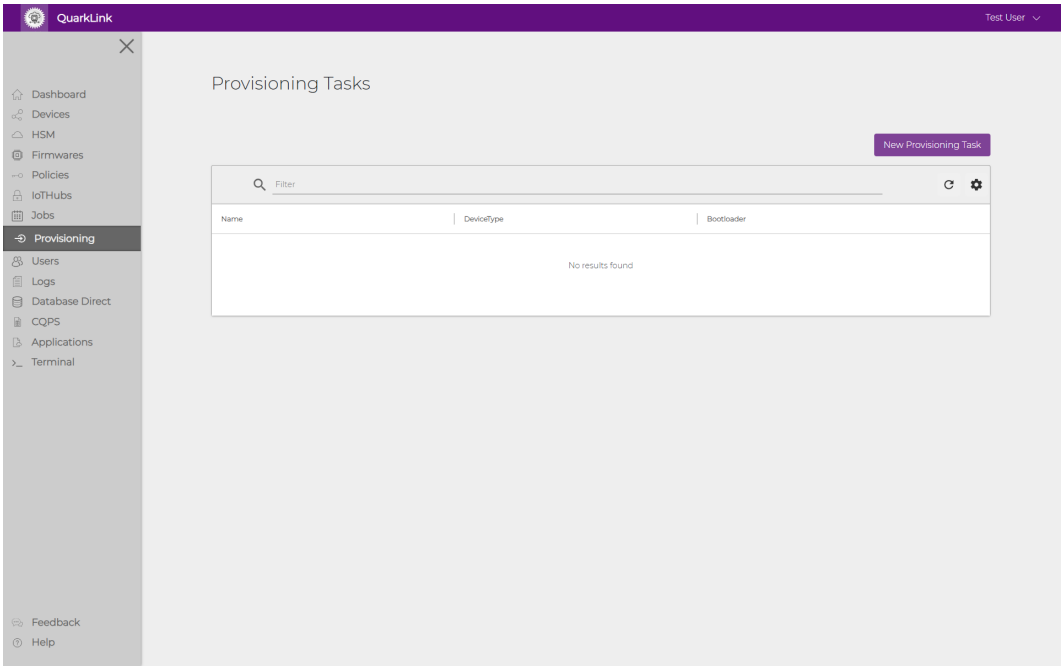


Figure 10: Navigating to the "Provisioning" section to create a new task

6. In the new window, choose a name for the provisioning task - here, we'll use "cordelia-provisioning-task."

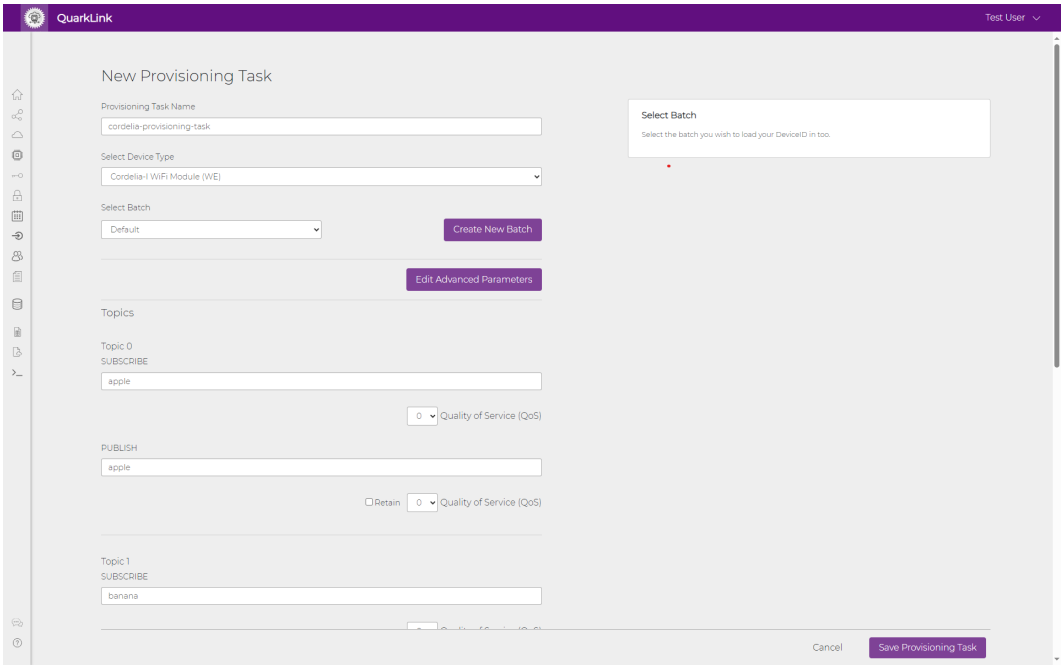


Figure 11: Naming the new provisioning task

7. For the device type, select "Cordelia-I WiFi Module (WE)." For the batch, you can select the "Default" option.

8. Click "Edit Advanced Parameters" to set up parameters for the Certificate Signing Request (CSR) and additional MQTT parameters. Default values are pre-filled for all fields, but you should use your own information. For example, update the country code from "DE" to your actual country code. The same applies for state, locality, surname, email, organization, and unit.

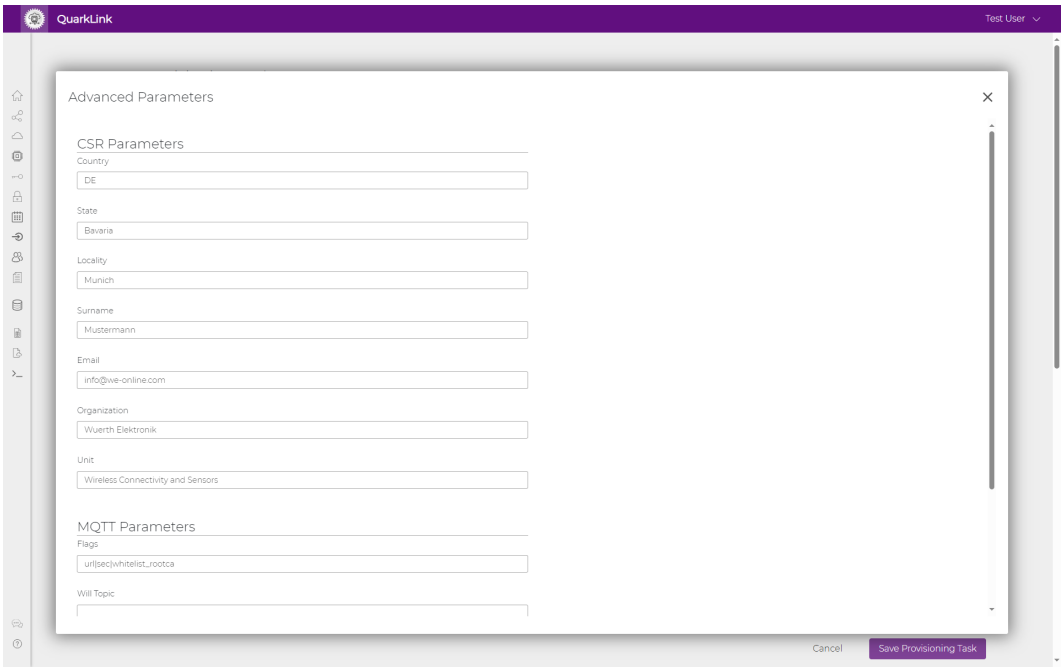


Figure 12: Editing advanced parameters for CSR and MQTT configurations

9. In the "Edit Advanced Parameters" window, leave the MQTT Parameters at their default values, then click "Save" to save your settings.

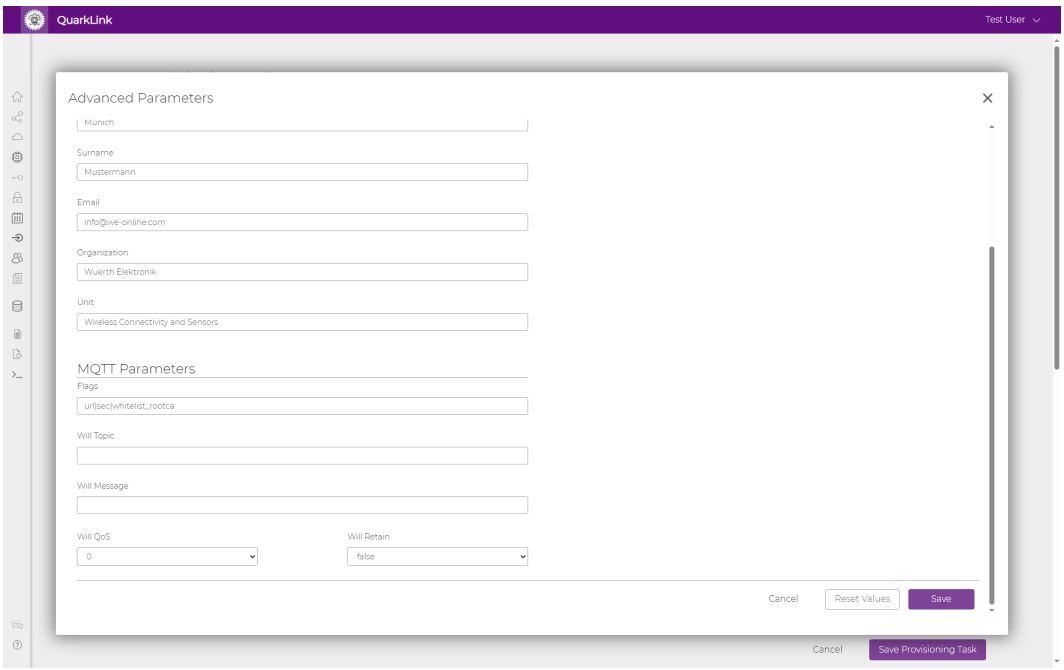


Figure 13: Saving the advanced parameters for provisioning

10. By default, the four subtopics and pubtopics (the MQTT topics that the module automatically subscribes to and the four topics it can publish to) are set with some placeholder values. We'll use these default values, so no changes are needed here.

11. Click "Save Provisioning Task" to save your provisioning task and parameters. You can rerun this provisioning task as often as you like for your Cordelia-I modules without needing to re-enter parameters each time.

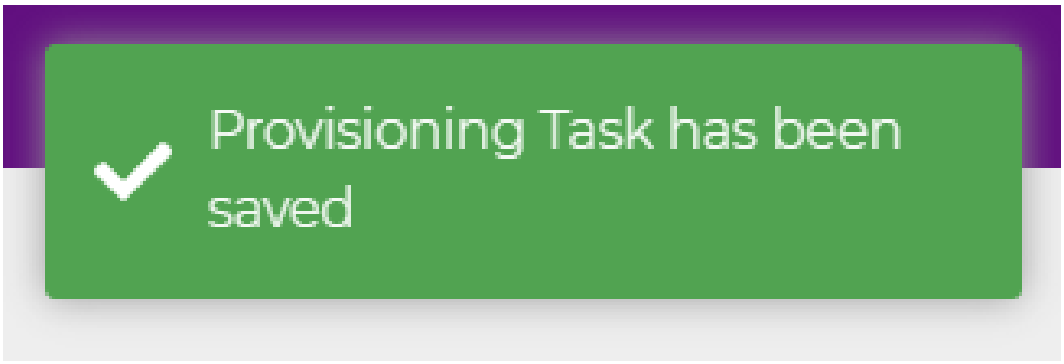


Figure 14: Saving the new provisioning task with defined parameters

12. Now the exciting part begins! Connect your Cordelia-I evaluation board to your PC with a USB cable. By default, the device starts in AT command mode and uses a virtual COM port (115200 baud, 8N1) for communication with the PC.

13. In the Provisioning menu, at the rightmost side of your newly created provisioning task, click the "Run Provisioning Task" button.

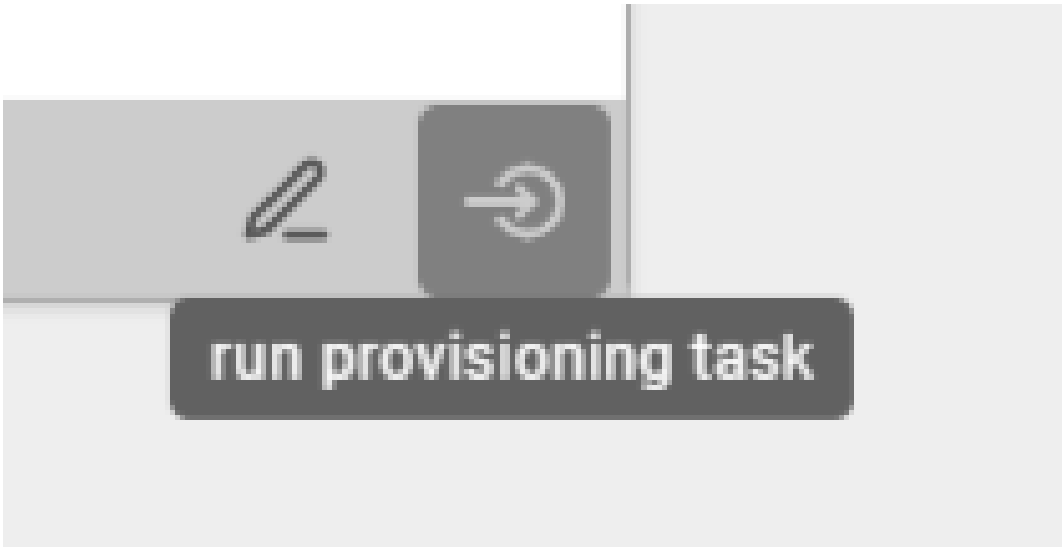


Figure 15: Running the provisioning task from the QuarkLink portal

14. In the upcoming window, enter your WLAN SSID and password. The Cordelia-I module will use this information to connect to the WLAN network.

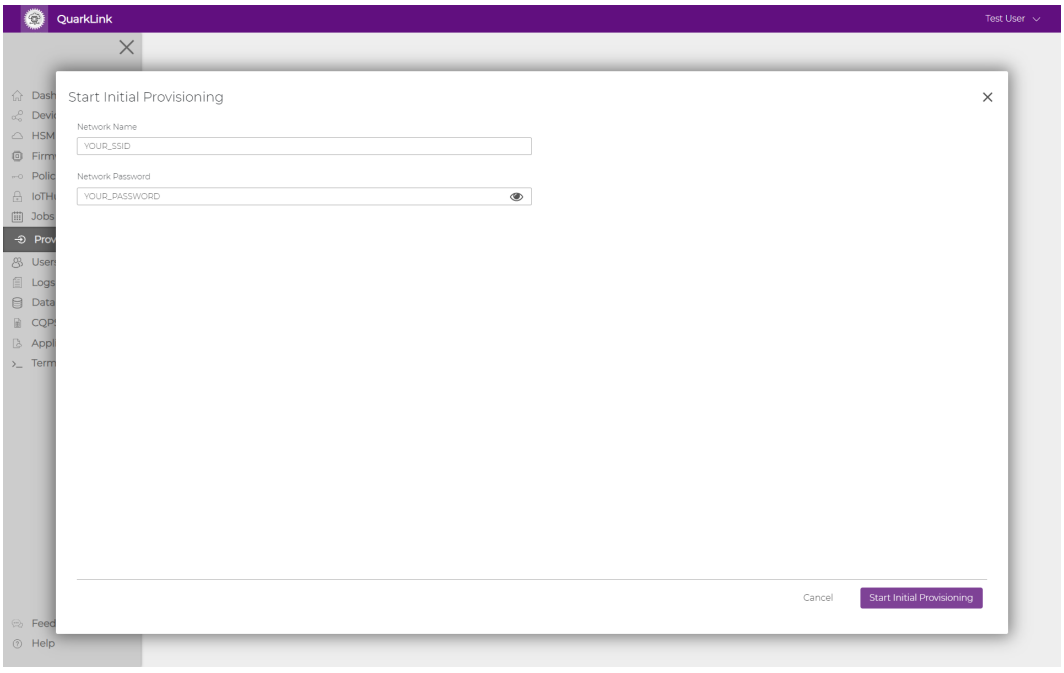


Figure 16: Entering WLAN credentials for the device's Wi-Fi connection

15. Your browser will prompt you for permissions to use the virtual COM port for serial communication. Select the COM port where your Cordelia-I module is connected and allow permissions.

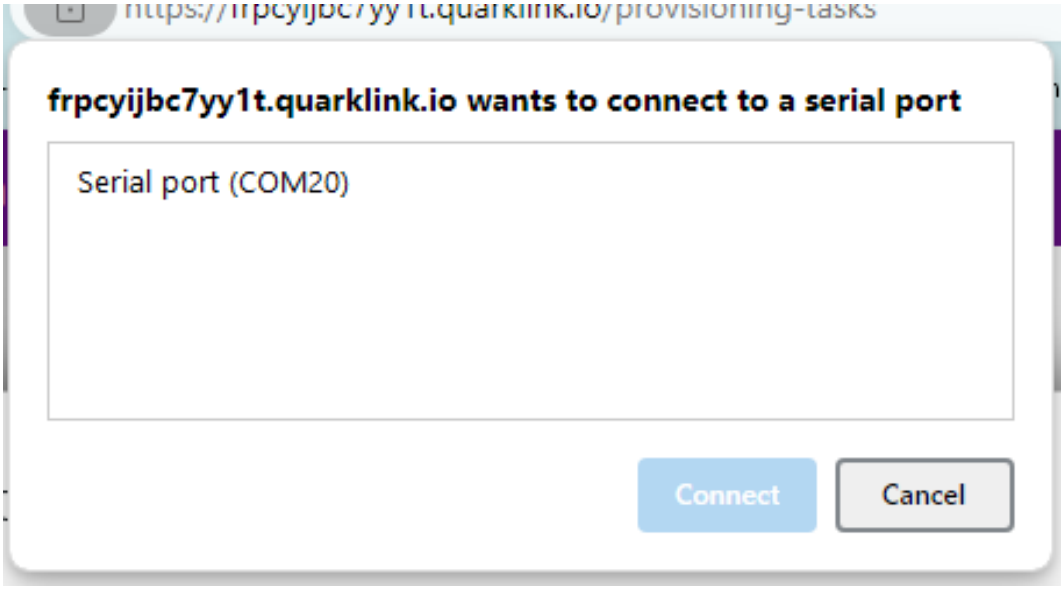


Figure 17: Granting permissions for the virtual COM port connection

16. The QuarkLink portal will now start the automatic provisioning process. You can follow the status of the process in the window. When it finishes, you should see a message indicating that your device has been provisioned and added to the Default batch. Click "Close" to exit this window.

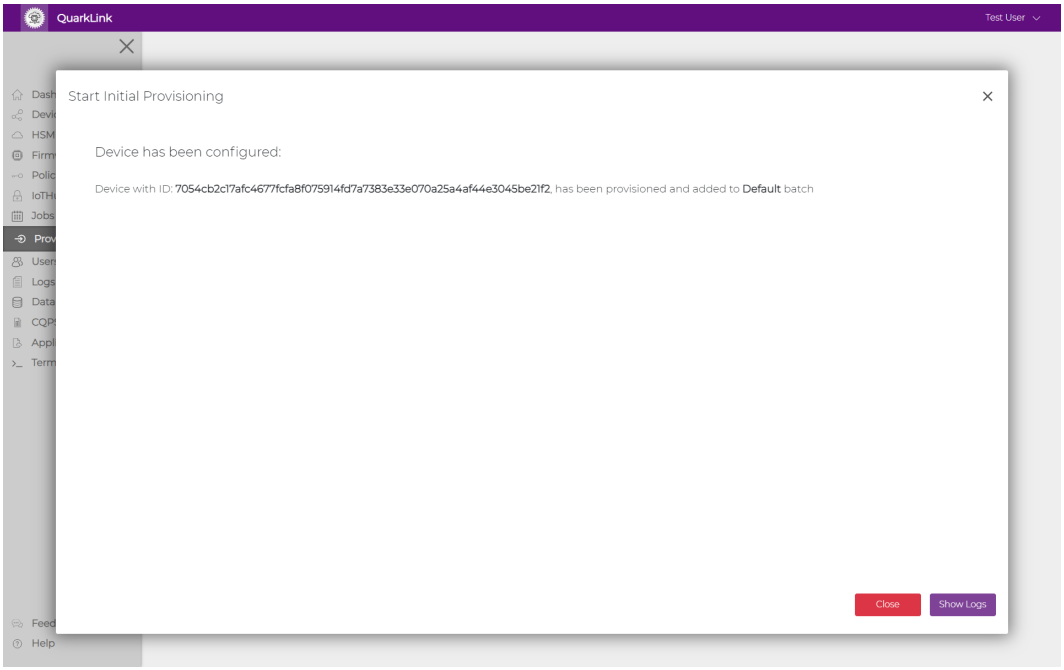


Figure 18: Monitoring the automatic provisioning process in QuarkLink

17. Now, click the menu item "Terminal." This will open a serial terminal emulator in your browser.

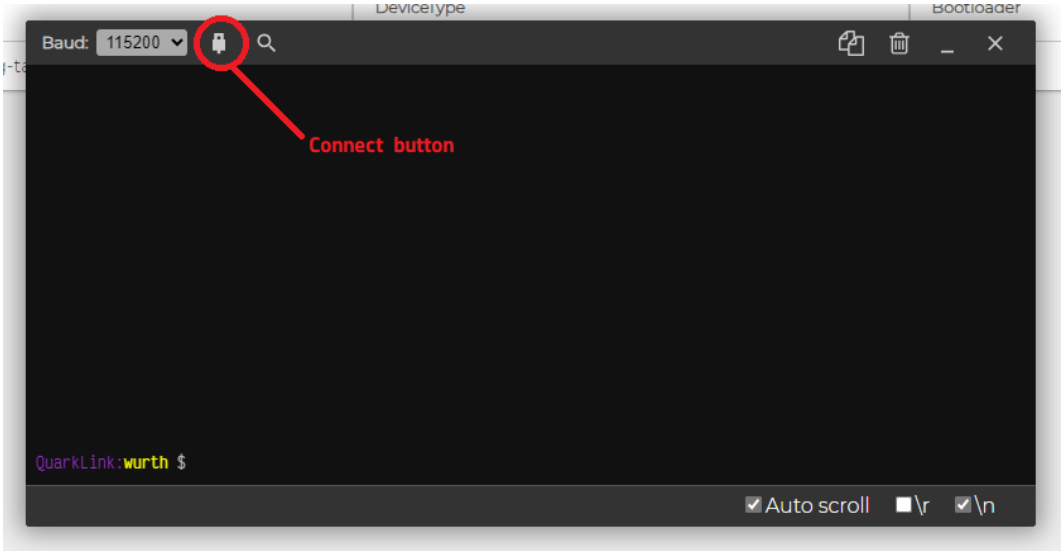


Figure 19: Opening the serial terminal emulator in the QuarkLink portal

18. Leave the baud rate at the default value of 115200, then click the "Connect port" button. Your browser may again ask for permission to use the virtual COM port. Select the COM port connected to your Cordelia-I module and grant permission.

19. If the connection is successful, the "Connect port" button will turn green, indicating that you are now ready to communicate with the module via AT commands!

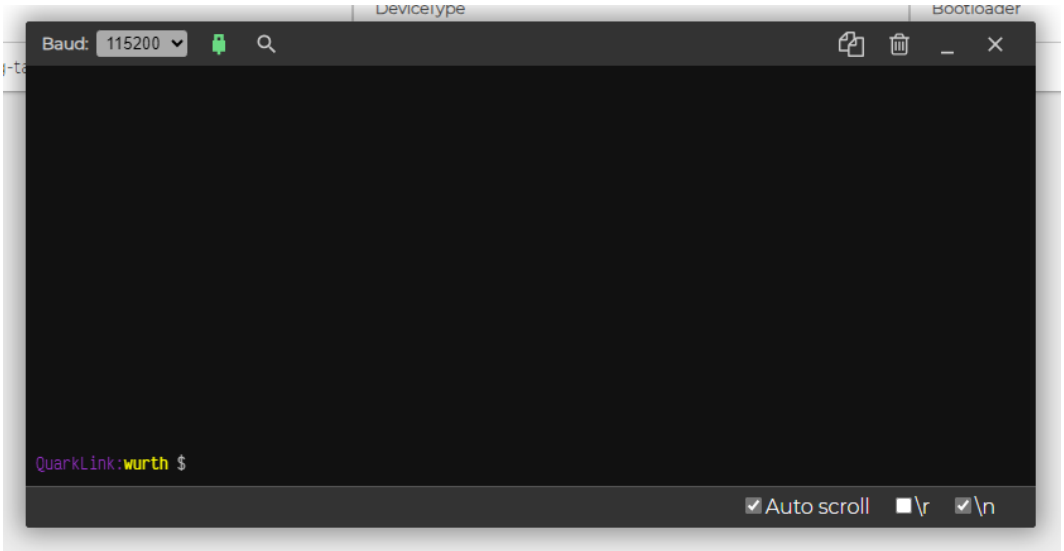


Figure 20: Connecting to the module's COM port in the terminal emulator

20. In the serial terminal, type `AT+iotenrol` and press Enter (you might have to press Enter twice). This starts the module's enrolment process, which activates it for secure data transmission and sets up all necessary security assets on the Cordelia-I module. This step is essential, because all the assets and neccessary user settings for the secure connection are being set during this process.

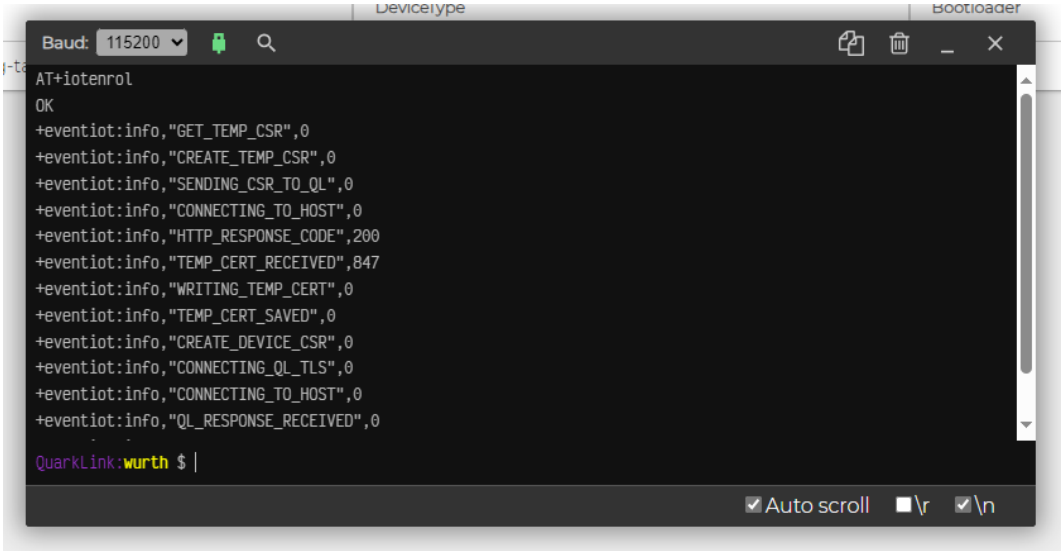


Figure 21: Starting module enrolment using the AT+iotenrol command

21. If enrolment is successful, you should see a message like: `+eventiot:info,"ENROLMENT_COMPLETE",0`. If you see an error message, check your configuration parameters and try running the provisioning task again.

22. In the serial terminal, type `AT+iotconnect` and press Enter. This will make the module connect to the secure MQTT broker configured on the QuarkLink portal.

23. If the connection is successful, you should receive a message like: `+eventmqtt:info,"CONNACK",0`.

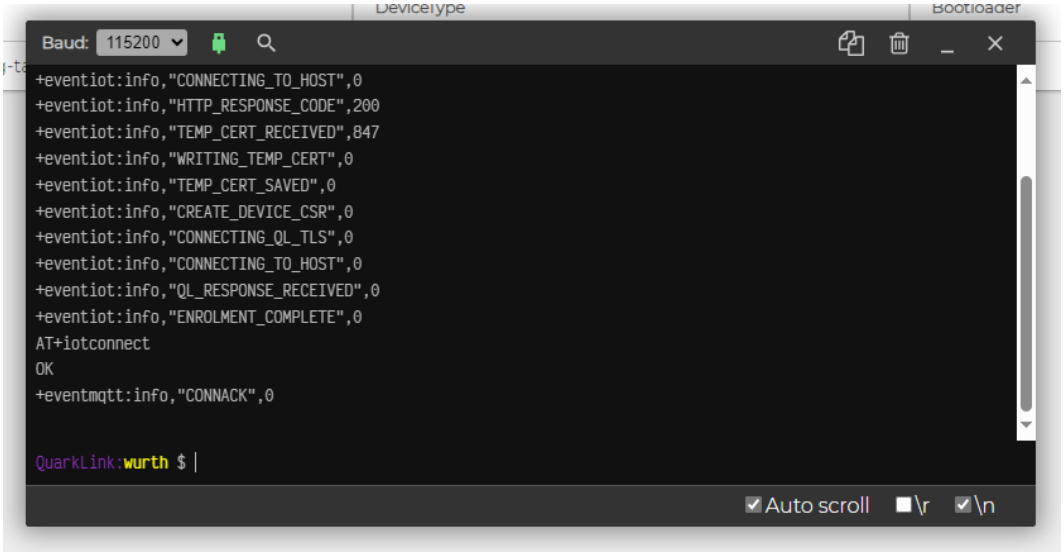


Figure 22: Connecting the module to the secure MQTT broker

24. Great! You've established a secure MQTT channel between your Cordelia-I module and the secure MQTT broker. You're now ready to send and receive messages on this secure channel.

25. Remember, in the provisioning task, we left the subtopics and pubtopics (the MQTT topics the module subscribes to and the topics it can publish to) at their default values. The default values are the same for the four subtopics and pubtopics (e.g., "apple," "banana," "orange," and "kiwi"). This means that after a successful connection, the Cordelia-I module automatically subscribes to these topics. So, if we publish something to any of these topics, we should receive it on our Cordelia-I module. Let's give it a try!

26. In the serial terminal, type `AT+iotpublish=1,"Hello Cordelia!"` (note that the message must be enclosed in quotation marks, and the maximum length of the payload can be 1011 bytes) and press Enter. Since we set the pubtopics to match the subscribed subtopics, you should see the message appear on the subscribed topic. In this case, we're publishing to index 1, which corresponds to "banana."

27. In the serial terminal, you should see an event showing an incoming MQTT message on the "banana" topic, which looks like this:
`+eventmqtt:recv,banana,qos0,"Hello Cordelia!"`.
This means the secure MQTT communication was successful. You can change the index in the `AT+iotpublish=1,"Hello Cordelia!"` command from 1 to any number between 0 and 3 to publish to other topics. In the provisioning task setup, you can also set any subtopics and pubtopics you like (leaving some empty if you wish). You can update these topics anytime in the QuarkLink portal under the Provisioning menu.

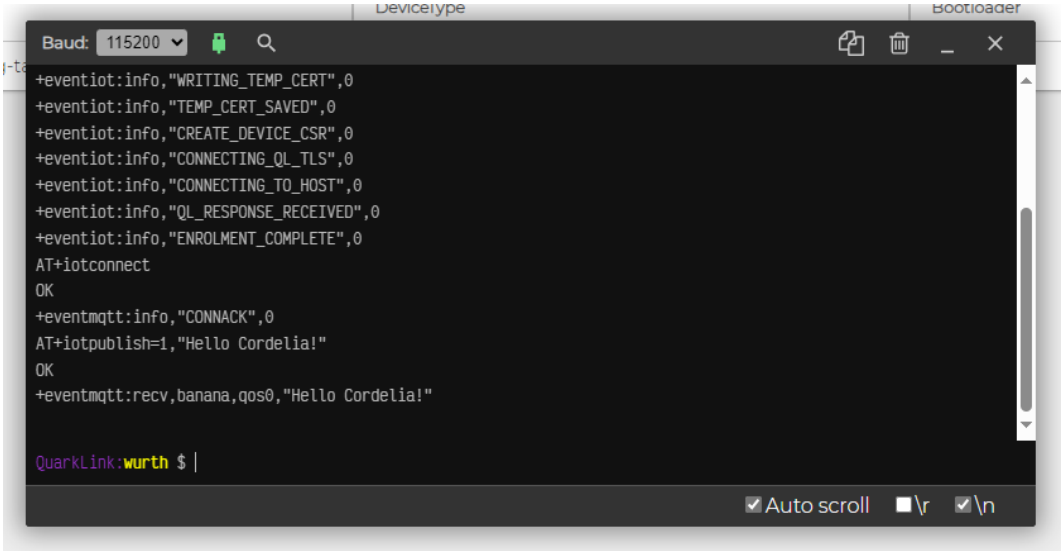


Figure 23: Sending a test message on a secure MQTT channel

Congratulations! You've just successfully sent and received data using secure MQTT communication. Feel free to experiment with the parameters and customize them for your application! For your reference, everything QuarkLink accomplished during the automated provisioning process can also be done manually using a classic serial terminal emulator and AT commands (you can find some examples for that in the "Use cases and examples" section). However, in that case, you would need to manually upload certificate files and configure each user setting for a secure connection. Using the QuarkLink portal simplifies this setup process and provides a proven, secure method for configuring your Cordelia-I module.
Thank you for following along, and enjoy exploring the capabilities of your Cordelia-I!

8 Cybersecurity

The Cordelia-I module is intended to provide a secure cloud connectivity interface to any embedded system. This chapter describes the security features of the Cordelia-I module and provides best practices to design and develop secure end-devices.

8.1 Security features

The Cordelia-I is built on a secure platform which provides several security features. Each of the major features will be presented in the following sections.

8.1.1 Secure file storage

The module has a secure file system encrypted by a unique private key per physical device. The File system is placed on an external Flash IC. Due to the nature of the unique key, even with access on the Flash IC itself over SPI the flash cannot be cloned or decrypted. The private key is not readable by the firmware of the radio module and therefore not by anybody.

The secure file system provides a place to store information such as certificates and private keys. The files themselves can be secured further with a token. Access to these files can be set as per the requirements of the end application (Read protection, write protection etc.) in compliance to cybersecurity best practises.

The files that are used by the Cordelia-I application (certificates) are configured as fail-safe and read-only.

8.1.2 Secure boot

The application firmware that runs on the IC is signed using a private key from Würth Elektronik eiSos. The corresponding public key is stored in the file system of the module. On each boot up, the bootloader verifies the integrity of the installed firmware by checking the signature using the stored public key. The images that do not match are not allowed to boot up.

8.1.3 Secure FOTA

The module supports a secure FOTA (See section 16). Depending on the server configuration, the module connects to the update server over TLS1.2. The update packet itself is signed using a private key. This signature is used to check the integrity of the downloaded packet before installing the same into the module. The update packet contains the following,

- Application firmware
- NWP ROM patch (Service pack)
- Root certificate catalog (Root CA catalog)

- Files that go into the file system
- Bootloader

The update process is failsafe. The module returns to the last verified working state if the update process fails or is interrupted. A roll back to a previous version after a successful update is blocked. Performing the factory reset action will revert the firmware version and memory contents to the production state.

The module does not automatically update the firmware or automatically checks for avialability of a new firmware. This must be triggered by the host MCU of the end system.

8.1.4 Secure Root of Trust

Each module comes with a unique pre-installed and hardware tamper-proof set of keys. The private key is fixed to the hardware and cannot be read out by the application. It can be used directly by the NWP. The application software can access only the public key and can generate further key pairs for cryptographic operations.

The module derives its unique device ID from this pre-installed private key and hence provides a secure root of trust to the end-device.

8.1.5 Network security

The Cordelia-I module supports WPA3 authentication mechanism on the WLAN interface and TLSv1.2 at the transport level. A combination of these protocols enables a robust secure network connection over the internet.

8.1.6 X.509 based PKI

For the socket layer security, the module supports the standard X.509 PKI based authentication and encryption methods.

8.1.7 Root CA Certificate catalog

The module comes with a list of trusted root CAs. During the server verification process, the module checks against this list to verify the chain of trust. This root CA catalog is issued by the IC manufacturer (Texas Instruments) and can be updated via FOTA.

8.2 Security design guide

This section describes the guidelines that enable easy integration of the Cordelia-I module in to an end application keeping in mind the cybersecurity requirements of the end-application.

- When possible, use the remote provision method (see section 6.7) for provisioning the end-device. This is the most secure and most scalable method to configure the device.
- If the end product intends to use the local provisioning mode (see section 12), it is mandatory to make sure that the password corresponding to the SoftAP of the module shall be set as per the NIST special publication 800-63B [i.3] before the provisioning is started. It is necessary to ensure that the passwords are unique and sufficiently strong.
- If the end product intends to use the local provisioning mode (see section 12), it is mandatory to make sure that the provisioning page implements system use notification messages as per IEC 62443-4-2: CR 1.12.
- The end device shall ensure that the module is updatable and regularly updated sing the FOTA mechanism (see section 16). To this end, the end device shall ensure that the module is able to boot-up in FOTA mode and the usersettings for the FOTA update are correctly set.
- The module outputs events in the format specified in the user manual. The end device shall ensure proper audit logging as well as storage mechanisms as per IEC 62443-4-2: CR 2.8 - 2.11.
- To create true end-to-end security (host via WLAN module to Cloud) a user specific encryption layer on the payload data can be implemented.
- The end application shall restrict the physical acccess to the components inside the end appliation - to be application agnostic the UART inbetween Cordelia-I and host is not encrypted.

8.2.1 Information on availability of product updates

There is a dependency on the IC manufacturer Texas Instruments to provide updates for the Chipset and Patches for the ROM content of the IC. Once Würth Elektronik eiSos gets the information that updates are present, we will check if the patched items apply to our application and criticality is estimated. If that is the case a firmware update will be queued into the corresponding teams to be released to the customer as FOTA package.

Once the update is ready for release the JEDEC compliant PCN process will be started. Depending on criticality with a reduced timespan. All our active customers will be informed by the PCN using the criteria defined by the company (i.e. active customers only). It is necessary on the customer's side to forward the information to the corresponding teams.

Please check with your direct sales contact that the PCN email addresses are maintained in the CRM system.

8.2.2 Vulnerability reporting

Please report any vulnerabilities when detected to *WCS@we-online.com*. The technical support team will guide you through the further process.

9 Host connection

The Cordelia-I is intended to be used as a radio module in a system, interfaced with a host micro-controller. The use of 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 a 8051 to the more advanced ARM core architecture.

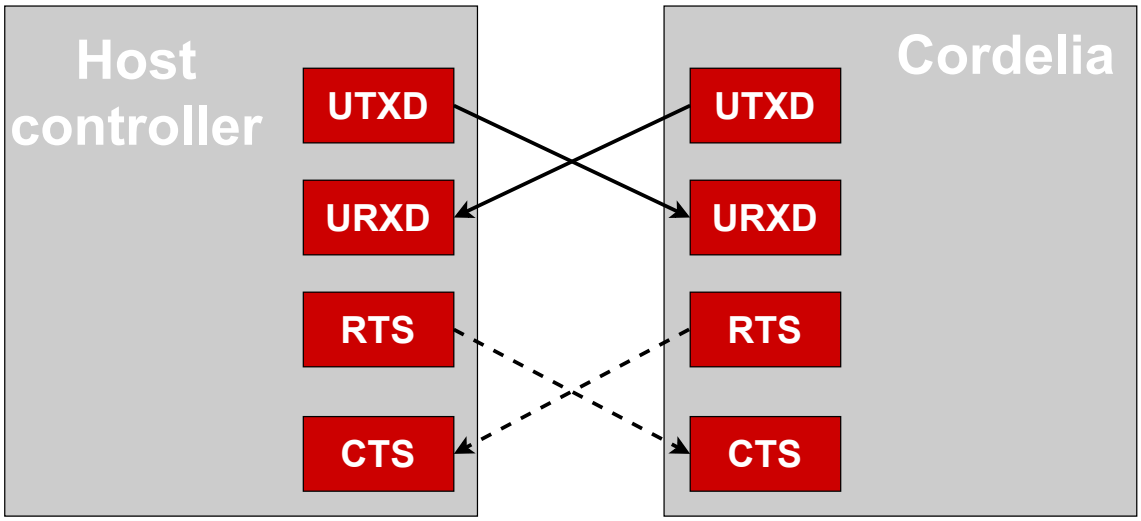


Figure 24: Host interface

9.1 UART parameters

The Cordelia-I implements the standard UART interface with the following parameters.

Parameter	Range	Standard
Baud	115200 to 3000000	115200
Data bits	8	8
Stop bits	1	1
Parity	none, odd, even	none
Flow control	none, RTS/CTS	none

Table 18: UART parameters

The configuration of the UART in factory state is 115200 baud with data format of 8 data bits, no parity and 1 stop bit ("8n1"). The baud rate, parity and flow control of the UART can be configured using the corresponding commands (see section 11.1). The data format is fixed to 8 data bits and one stop bit. This results in a user data ratio of 11 UART symbols per 8 bit.

9.2 Hardware flow control

Hardware flow control is disabled by default. It is recommended not to use baud rates higher than 921600 baud if flow control is disabled.
In case flow control is enabled by using the `AT+set` command baud rates of up to 3 MBaud are supported.

9.3 Timing and characteristics

The output of characters on the serial interface runs with secondary priority. For this reason, short interruptions may occur between the output of successive bytes. The host must not implement a strict timeout between two bytes to be able to receive packets that have interruptions in between. Up to four full byte durations (32 bit) delay between two successive bytes shall be accepted by the host.

For the direction "host to module", the module also accepts a pause of up to four full byte durations (32 bit) delay between two successive bytes before discarding received content (without user notification).
Additionally, in order to ensure proper processing of the AT commands, a short guard interval is necessary between the receipt of a confirmation/indication and the host sending the next command. As shown in figure 25, the guard interval (t4-t3) must be at least 40 µs.

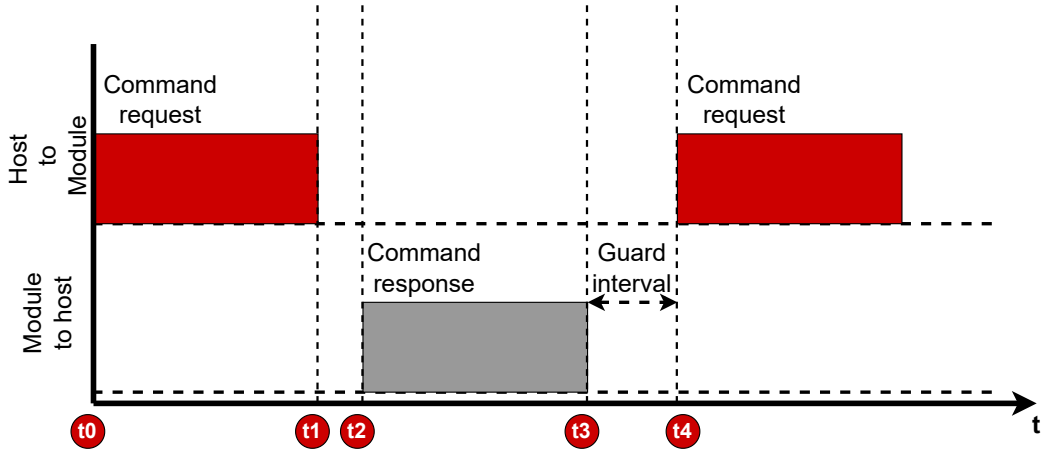


Figure 25: UART timing

10 The command interface

The command interface on the Cordelia-I enables full control over the module using ASCII based AT styled commands, followed by a "\r\n" (hex: 0x0D0A).

10.1 Command types

There are three types of messages exchanged between the Cordelia-I and the host.

- Requests: The host requests the module to perform an action or start an operation.
  All requests start with the "AT+" tag and end with "\r\n" (hex: 0x0D0A).
- Confirmations: On each request, the module answers with a confirmation message to give a feedback on the requested operation status.
  All confirmations contain the request itself and either a "OK" or an error code. Appendix 31 gives a brief description of all the error codes. All confirmations end with "\r\n" (hex: 0x0D0A).
- Events: The module indicates spontaneously when a special event has occurred. All events start with the "+" tag and contain further data, error codes (see Appendix 31) or status information. All events end with "\r\n" (hex: 0x0D0A).

10.2 AT command characteristics

This section describes the syntax and detailed characteristics of the aforementioned three command types.

10.2.1 Request

The generic syntax of an AT command request is as shown below :

```
AT+<command name> = <param1>, <param2>, ..., <paramX>
```

- All commands start with the prefix "AT". The delimiter "+" indicates the beginning of the command name.
- Commands can have parameters in which case the delimiter "=" separates the command name from the list of parameters.
- AT commands can be entered in upper or lower case
- It is recommended to use enclosing quotation marks when providing string parameters.
- Furthermore, each parameter is separated from the next with a "," (comma) delimiter. A comma shall not be preceeded or followed directly by a whitespace
- In cases where a parameter is optional or ignored, it may be left empty. Nevertheless the "," delimiter has to be present. An empty parameter in a AT command looks like ",,".
- String parameters containing spaces must be enclosed with quotation marks ("").

- All hexadecimal parameters must have a 0x prefix.
- MAC and network addresses must be entered as follows
  - MAC address - Six hexadecimal values of 8 bit each, represented as X:X:X:X:X:X (X can range from 0x00 up to 0xFF), the ":" is used as delimiter
  - IPv4 address - Four decimal values of 8 bit each, represented as X.X.X.X (X can range from 0 up to 255), the "." is used as delimiter per 8 bit
  - IPv6 address - Four hexadecimal numeric values of 32 bit each, represented as X:X:X:X (X can range from 0x00000000 up to 0xFFFFFFFF), the ":" is used as delimiter per 32 bit
- Bit-mask parameters are represented using "|" delimiter for concatonation of multiple paramters - e.g. x|y|z when the parameters x and y and z are to be set.
- Data should be either binary or Base64 format (binary to text encoding). Further details about Base64 data encoding including reference implementation can be found in the RFC 4648 of the IETF (Internet Engineering Task Force).

10.2.2 Confirmations

The command confirmations have the following syntax,

```
<command name>:<value1>, <value2>, ..., <valueX>
```

On success, the confirmations contain a positive acknowledgement.

```
OK
```

In case of an error, the corresponding error code and an optional description is returned.

```
ERROR:<error description>, <error code>
```

10.2.3 Events

Asynchronous events can arrive at any time and are formatted as follows.

```
+<event name>:<value1>, <value2>, ..., <valueX>
```

10.2.4 Help

The AT command interface has a built-in quick help feature. On sending a "?" character instead of parameter list, the Cordelia-I responds with a list of parameters that are expected for the corresponding command.

```
AT+<command name> = ?
<param1>, <param2>, ..., <paramX>
OK
```

For example

```
AT+wlanConnect=?
[SSID],[BSSID],[SecurityType],[SecurityKey],[SecurityExtUser],[SecurityExtAnonUser],[
    SecurityExtEapMethod]
OK
```

11 AT commands

In this chapter, various commands used to configure and control the Cordelia-I module are described.
The AT command set is based on ASCII coding of any data. Unless the command requires explicitly a different coding than ASCII.

11.1 Device commands

The commands in the device category provide access to generic module properties like communication interface, time and date settings and version information. Additionally, basic device operations like start, stop, reboot and sleep are described in this section.

11.1.1 Start and stop commands

The start and stop commands control the state of the 802.11 network processor unit (NWP). On boot up the network processor is started by default. A stop command puts the network processor to hibernate effectively switching off the radio resulting in loss of all on-going transmissions and connections. A time-out can be specified to allow the network processor to gracefully disconnect before shutting down.

Request	Response
AT+start	OK or error
Arguments: None	

Table 19: AT+start

Request	Response
AT+stop=[timeout]	OK or error
Arguments: timeout: in milliseconds • 0 - Stop immediately without waiting for a response from the NWP. • 65535 - Wait indefinitely for a response from the NWP. • 0 < timeout < 65535 - Wait for timeout before forcing the NWP to stop.	

Table 20: AT+stop

11.1.2 Test

This command provides a simple way of ensuring that the module is active and ready to receive further commands.

Request	Response
AT+test	OK or error
Arguments: None	

Table 21: AT+test

11.1.3 Reboot

This command performs a software reset on the module. The module internally puts the NWP to hibernate before restarting from the reset vector.

Request	Response
AT+reboot	OK or error
Arguments: None	

Table 22: AT+reboot

It is recommended to use this command whenever possible instead of a hard reset (a falling edge on the */Reset* pin).

11.1.4 Factory reset

The factory reset command restores the module to factory state.

- All files stored in the file system will be reverted to factory state.
- New files that were added will be deleted.
- The network processor settings including MAC address will be restored to factory state.

Request	Response
AT+factoryreset	OK or error
Arguments: None	

Table 23: AT+factoryreset

Factory reset operation can take up to 90 seconds to complete. The module responds with an "OK" only after this time period. A start-up message after the "OK" indicates the completion of the factory reset operation.

Resetting or powering off the module during this operation can result in permanent damage to the module.

A reset is performed automatically after the restore operation.

11.1.5 Hibernate

The sleep command puts the module into the lowest possible power mode (hibernate) resulting in a current consumption of less than 10 µA. In hibernate mode, the network processor is in hibernate mode and the application processor is shut down.
The module wakes up automatically after a time period specified in the sleep command. Alternatively, the module can be woken up manually with a rising edge on the *WAKE_UP* pin. On any wake up trigger, the module starts from the reset vector.

Request	Response
AT+hibernate=[timeout]	OK or error
Arguments: timeout: in seconds • 0 - sleep forever. • 1 <= timeout <= 86400 - Wait for timeout seconds before wake-up.	

Table 24: AT+sleep

11.2 Module parameters Get/Set

11.2.1 Get

The generic get command can be used to read the device parameters including version, time, UDID, UART and transparent mode settings. The system persistent setting is enabled by default. This means that all the settings are retained after reset.
A detailed description of the parameters that can be configured is described in section 6.2.

Request		Response
AT+get=[ID],[option]		+Get:[value1],. . . ,[valueX] OK or error
Arguments:		Arguments:
ID	**option**	**value1, . . . , valueX**
general	version	value1: chip ID value2: MAC version (X.X.X.X) value3: PHY version (X.X.X.X) value4: NWP Version (X.X.X.X) value5: ROM version (X)) value6: Cordelia-I FW version (X.X.X)
	time	value1:hh, value2:mm, value3:ss, value4:dd, value4:mm, value6:yyyy
	persistent	value1:0 or 1 (1=enable, 0=disable)
IOT	UDID	value1:16 byte UDID (unique device identifier)
	deviceid	value1:64 byte device ID derived from the device unique private key
UART	baudrate	value1:baudrate [Baud] (see chapter 9.1)
	parity	value1:0, 1 or 2(0=none, 1=even, 2=odd)
	flowcontrol	value1:0 or 1 (1=enable, 0=disable)
	transparent_trigger (see chapter 13)	value1:bitmask -timer -1etx -2etx -transmit_etx example for value1: timer|2etx
	transparent_timeout	value1:timeout in [ms], range 6-1000
	transparent_etx	value1:2 byte ETX (hex), e.g. 0x0D0A
CSR	country	value1:Country code string
	state	value1:State/Province string
	locality	value1:City/Town or Locality string
	surname	value1:Surname string
	email	value1:E-mail string
	organization	value1:Organization string
	unit	value1:unit string
QUARKLINK	hostname	value1:Address of the QuarkLink instance
	port	value1:port number (Typical:6000)
	rootCAPath	value1:path to the rootCA file of the QL server

Table 25: AT+get (Part 1)

Request		Response
AT+get=[ID],[option]		+Get:[value1],. . . ,[valueX] OK or error
Arguments:		Arguments:
ID	**option**	**value1, . . . , valueX**
OTA	url	value1:Address of the FOTA server
	rootCAPath	value1:path to the rootCA file of the OTA server
	updateVersion	value1:Version of the update firmware available
MQTT	iotHubEndpoint	value1:Address of the MQTT end point
	iotHubPort	value1:port number
	rootCAPath	value1:path to the rootCA file of the MQTT end point
	clientCertPath	value1:path to the client certificate path
	clientPrivateKey	value1:path to the client private key
	clientId	value1:ID of the MQTT client
	flags	value1:One or more of the following values separated by "|". Permitted flags - ip4, ip6, url, sec, skip_domain_verify, skip_cert_verify, skip_date_verify
	base64	value1:0 or 1 (1=Payload is base64 encoded; 0= Payload is binary)
	willTopic	value1:Will topic string
	willMessage	value1:Will message payload
	willQos	value1:0, 1 or 2 (0=QoS0, 1=QoS1, 2=QoS2)
	willRetain	0=do not retain, 1=retain
	userName	value1:Username string
	password	value1:Password string
	keepAliveTimeOut	value1:Keep alive timeout
	cleanConnect	value1: 0 or 1 (0=false, 1=true)
SUBTOPIC**X** (**X** = 0-3)	name	value1:Topic string
	qos	value1:0, 1 or 2 (0=QoS0, 1=QoS1, 2=QoS2)
PUBTOPIC**X** (**X** = 0-3)	name	value1:Topic string
	retain	value1:0 or 1 (0=do not retain, 1=retain)
	qos	value1:0, 1 or 2 (0=QoS0, 1=QoS1, 2=QoS2)

Table 26: AT+get (Part 2)

11.2.2 Set

The generic set command can be used to set device parameters like time, persistence, UART and transparent mode settings.

Request		Response
AT+set=[ID],[option],[value1],. . . ,[valueX]		OK or error
Arguments:		
ID	**option**	**value1, . . . , valueX**
general	persistent	1=enable, 0=disable
	time	hh,mm,ss,dd,mm,yyyy (without preceding zeros)
UART	baudrate	baudrate [Baud] (see chapter 9.1)
	parity	0=none, 1=even, 2=odd
	flowcontrol	true, false
	transparent_trigger (see chapter 13)	value1: bitmask -timer -1etx -2etx -transmit_etx example for value1: timer|2etx
	transparent_timeout	value1: timeout in [ms], range 6-1000
	transparent_etx	2 byte ETX (hex), e.g. 0x0D0A
CSR	country	Country code string
	state	State/Province string
	locality	City/Town or Locality string
	surname	Surname string
	email	E-mail string
	organization	Organization string
	unit	unit string
QUARKLINK	hostname	Address of the QuarkLink instance
	port	port number (Typical:6000)
	rootCAPath	path to the rootCA file of the QL server
OTA	url	Address of the FOTA server
	rootCAPath	path to the rootCA file of the OTA server

Table 27: AT+set (Part 1)

Request		Response
AT+set=[ID],[option],[value1],. . . ,[valueX]		OK or error
Arguments:		
ID	**option**	**value1, . . . , valueX**
MQTT	iotHubEndpoint	Address of the MQTT end point
	iotHubPort	port number
	rootCAPath	path to the rootCA file of the MQTT end point
	clientCertPath	path to the client certificate path
	clientPrivateKey	path to the client private key
	clientId	ID of the MQTT client
	flags	One or more of the following values separated by "|". Permitted flags - ip4, ip6, url, sec, skip_domain_verify, skip_cert_verify, skip_date_verify
	base64	1=Payload is base64 encoded; 0= Payload is binary
	willTopic	Will topic string
	willMessage	Will message payload
	willQos	0=QoS0, 1=QoS1, 2=QoS2
	willRetain	0=do not retain, 1=retain
	userName	Username string
	password	Password string
	keepAliveTimeOut	Keep alive timeout
	cleanConnect	0=false, 1=true
SUBTOPIC**X** (**X** = 0-3)	name	Topic string
	qos	0=QoS0, 1=QoS1, 2=QoS2
PUBTOPIC**X** (**X** = 0-3)	name	Topic string
	retain	0=do not retain, 1=retain
	qos	0=QoS0, 1=QoS1, 2=QoS2

Table 28: AT+set (Part 2)

It is not possible to set MQTT parameters during an active MQTT connection.

11.3 WLAN commands

In this section, all the commands necessary to configure the WLAN settings of the module are described.

11.3.1 Scan

The scan function enables the user to perform a scan and discover devices on all the enabled channels. The module returns a list of up to 30 devices.

The first scan command initiates a scan and hence returns an error code SL_ERROR_WLAN_GET_NETWORK_LIST_EAGAIN (-2073). A further scan command returns the list of available access points.

Request	Response
AT+wlanScan=[index],[count]	+wlanscan:<Device[index]> . . . OK or error
Arguments: Index: starting index 0-29 count: number of devices, max. 30	Arguments: Each device has the following parameters listed SSID, BSSID, RSSI, Channel, Security type, hidden_ssid_enabled (0 or 1), cipher, key_management_method

Table 29: AT+wlanScan

11.3.2 Manual connection

In order to manually connect the Cordelia-I to a known access point, the following command has to be used. A manual connect has the highest priority over all the other connection types. A connect event confirms a successful connection.

Request	Response
AT+wlanConnect=[SSID], [BSSID], [SecurityType], [SecurityKey], [SecurityExtUser], [SecurityExtAnonUser], [SecurityExtEapMethod]	OK or error
Arguments: - SSID: Name of the AP - BSSID: MAC address of the AP (optional) - SecurityType: OPEN, WEP, WEP_SHARED, WPA_WPA2, WPA_ENT, WPS_PBC, WPS_PIN, WPA2_PLUS, WPA3 (see table 31) - SecurityKey: password (ignored if not applicable for selected SecurityType) - SecurityExtUser: Enterprise user name parameters (Ignored in case WPA_ENT was not selected) - SecurityExtAnonUser: Enterprise anonymous user name parameters (Ignored in case WPA_ENT was not selected) - SecurityExtEapMethod: Extensible Authentication Protocol (Ignored in case WPA_ENT was not selected): TLS, TTLS_TLS, TTLS_MSCHAPv2, TTLS_PSK, PEAP0_TLS, PEAP0_MSCHAPv2, PEAP0_PSK, PEAP1_TLS, PEAP1_PSK	

Table 30: AT+wlanConnect

The EAP methods with TLS supports TLS v1.0 only and not TLS v1.2

Type	Description	Password length
OPEN	No security	
WEP	WEP open security	13 or 26 characters
WEP_SHARED	WEP shared security	13 or 26 characters
WPA_WPA2	WPA-PSK and WPA2-PSK security types, or a mixed mode of WPA / WPA2-PSK security type (TKIP, AES, mixed mode)	8 to 63 characters
WPA2_PLUS	Supports connection to networks with security WPA3, WPA2+PMF (Protected Management Frames) and WPA2 (CCMP only)	8 to 63 characters
WPA3	Supports connection to WPA3 only networks	8 to 63 characters
WPA_ENT	Enterprise security	
WPS_PBC	WPS push-button security	
WPS_PIN	WPS pin code security	

Table 31: WLAN security types

A manual disconnect of an existing connection is done using the following command.

Request	Response
AT+wlanDisconnect	OK or error

Table 32: AT+wlanDisconnect

11.3.3 Profiles

Cordelia-I allows the user to store up to seven preferred networks as profiles. Based on the connection policy (see section 11.3.5) the module automatically establishes a connection using one of the saved profiles. Profile priority determines the order of connection. The profiles are saved in the non-volatile memory and can be added, read or deleted using the following commands.

Request	Response
AT+wlanProfileAdd=[SSID], [BSSID], [SecurityType], [SecurityKey], [SecurityExtUser], [SecurityExtAnonUser], [SecurityExtEapMethod],[priority]	+wlanProfileAdd: <Profile index> OK or error
Arguments: - SSID: Name of the AP - BSSID: MAC address of the AP (optional) - SecurityType: OPEN, WEP, WEP_SHARED, WPA_WPA2, WPA_ENT, WPS_PBC, WPS_PIN, WPA2_PLUS, WPA3 (see table 31) - SecurityKey: password (optional if not used) - SecurityExtUser: Enterprise user name parameters (Ignored in case WPA_ENT was not selected) - SecurityExtAnonUser: Enterprise anonymous user name parameters (Ignored in case WPA_ENT was not selected) - SecurityExtEapMethod: Extensible Authentication Protocol (Ignored in case WPA_ENT was not selected): TLS, TTLS_TLS, TTLS_MSCHAPv2, TTLS_PSK, PEAP0_TLS, PEAP0_MSCHAPv2, PEAP0_PSK, PEAP1_TLS, PEAP1_PSK - Profile priority: 0 - 15 (highest)	

Table 33: AT+wlanProfileAdd

Only one enterprise profile can be saved on to the non-volatile memory.

Request	Response
AT+wlanProfileGet=[index]	+wlanProfileGet:[value1], . . . , [value8] OK or error
Arguments: index: profile index, range 0 - 6	Arguments: value1 = SSID value2 = BSSID value3 = Security type value4 = Security key value5 = Security Ext User value6 = Security Ext Anon User value7 = Security EXT EAP method value8 = Priority

Table 34: AT+wlanProfileGet

Request	Response
AT+wlanProfileDel=[index]	OK or error
Arguments: index: profile index, range 0 - 6	

Table 35: AT+wlanProfileDel

11.3.4 WLAN settings

In this section commands to read and modify the WLAN settings in different modes are described. All the WLAN settings are non-volatile.

Request			Response
AT+wlanSet=[ID],[option],[value1],. . . ,[valueX]			OK or error
ID	**option**	**[value1],. . . ,[valueX]**	
general	COUNTRY_CODE	US (channels 1-11), EU (channels 1-13) or JP (channels 1-13)	
	STA_TX_POWER	0-15 (0 = Max transmit power)	
	SCAN_PARAMS	value1: channel mask (Channels bits order: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14) value2: RSSI threshold	
	SUSPEND_PROFILES	Suspend profile bit mask (Set bit n to suspend profile with index n)	
	DISABLE_ENT_SERVER_AUTH	0 or 1 (1 = disable server auth when manually connecting to an enterprise network)	

Table 36: AT+wlanSet

Request		Response
AT+wlanGet=[ID],[option]		+wlanGet:[value1],. . . ,[valueX] OK or error
Arguments:		Arguments: see table 36
ID	**option**	
general	COUNTRY_CODE	
	STA_TX_POWER	
	SCAN_PARAMS	

Table 37: AT+wlanGet

11.3.5 WLAN policy

This set of commands allows changes in behavior of the Cordelia-I with respect to connection, power consumption, scan as well as P2P connections.

- **Connection**: This policy defines how the device initiates and maintains a specific connection after reset. The following options are available (bit mask - one or more options can be set):

  Auto - The device automatically tries to connect to the stored profiles based on priority. In case of several profiles with the same priority, the decision is made based on security type (WPA2>WEP>OPEN). In case of the same security type, the one with the highest signal strength is chosen to be connected.

Fast - The device tries to connect to the last connected AP without transmitting a probe request.

P2P - The device connects to the first available WiFi direct device.

- **Scan**: Additional to the one-shot scan, Cordelia-I can be configured to perform periodic scans with a specific scan period.
- **Power management**: Based on the application, the power management policy of the WLAN NWP can be set to one of the following options: Normal, low latency, low power and long sleep.
- **P2P**: In P2P mode, the Cordelia-I can be configured to either choose a specific role (GO or client) or negotiate with the peer. The connection initiation can be active or passive based on the policy set.

Request			Response
AT+wlanPolicySet=[ID],[option],[value1]			OK or error
ID	**option**	**value1**	
connection	Auto, Fast or P2P (bit mask)		
scan	Hidden_SSID	scan interval in seconds	
	No_Hidden_SSID	scan interval in seconds	
	Disable_Scan		
PM	normal, low_latency, low_power or long_sleep	Maximum sleep time in ms only for long sleep option	

Table 38: AT+wlanPolicySet

Request	Response
AT+wlanPolicyGet=[Type]	+wlanPolicyGet:[option],[value1] OK or error
Arguments: connection, scan or PM	Arguments: (see table 38)

Table 39: AT+wlanPolicyGet

Supported Cipher methods
SSL_RSA_WITH_RC4_128_SHA
SSL_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256

Table 40: Supported cipher methods

11.3.6 Configure WiFi via AT command interface

To enter the credentials of the access point via AT commands, the Cordelia-I radio module must be started in "AT command mode" by applying a low level at the pins *APP_MODE_0* and *APP_MODE_1* during start-up.

If this has been done, the credentials can be added in two ways. Either to use the command `AT+wlanConnect` to connect to an access point, or to use the command `AT+wlanProfileAdd` to simply add the profile to the radio module.
Then the module automatically connects to one of the configured access points after device reset, in case the WLAN connection policy is set to `Auto` (or `Auto|Fast`).

```
AT+wlanPolicySet=connection,Auto|Fast,
```

Code 1: Example AT+wlanPolicySet

The policy is automatically set to `Auto|Fast` when the module is started in transparent mode.

AT+wlanConnect

```
AT+wlanConnect=[SSID],[BSSID],[SecurityType],[SecurityKey],[SecurityExtUser],[
    SecurityExtAnonUser],
[SecurityExtEapMethod]
```

Code 2: AT+wlanConnect

```
AT+wlanConnect=mySSID,,WPA_WPA2,myPassword,,,
```

Code 3: Example AT+wlanConnect

AT+wlanProfileAdd

The command `AT+wlanProfileAdd` allows you to store up to seven preferred WLAN profiles which you can use to specify multiple access points to be used in transparent mode. Each profile stores the access point credentials along with a profile priority, which determines the order of connection.

```
AT+wlanProfileAdd=[SSID],[BSSID],[SecurityType],[SecurityKey],[SecurityExtUser],
[SecurityExtAnonUser],[SecurityExtEapMethod],[priority]
```

Code 4: AT+wlanProfileAdd

The following example adds a profile with the highest priority (15).

```
AT+wlanProfileAdd=mySSID,,WPA_WPA2,myPassword,,,,15
```

Code 5: Example AT+wlanProfileAdd

11.4 File system commands

Cordelia-I creates and maintains an encrypted file system on the serial flash present on-board. The file system provides secure storage for files like certificates, private keys and web pages. In the following, some of the features of the file system are described.
The storage capacity for additional content in the radio module's file system is limited to the available capacity in the file system itself.

- The file system can only be accessed through AT commands.
- The file system on one module cannot be read by another - this prevents cloning of sFlash.
- Built in tamper detection detects corrupt files and warns the user of unauthenticated file access.
- Each file has a minimum size of 4096 bytes (Fail-safe = 8192 bytes).
- The maximum number of files is 240 of which 100 are reserved for system files.
- File names may have a maximum size of 180 characters.
- Files can be created with one or more of the following flags: fail-safe, secure, public read, public write.
- Files cannot be enlarged once created, hence the maximum size attribute has to be set appropriately during file creation.

Minimize the number of writes to flash to ensure data endurance.

A file creation/deletion updates the FAT table. Rewrite/overwrite files when possible.

Care needs to be taken to have a clean and stable supply voltage especially during flash writes in battery powered applications. A drop in voltage during an erase cycle may lead to corruption of the file system.

11.4.1 File system operations

Request	Response
AT+fileGetFileList	+FileGetFileList:[fileName],[maxFileSize], [properties],[fileBlocksAlloc] OK or error
Arguments: None	fileName: File name maxFileSize: Max file size properties: Bit mask - open_write - open_read - must_commit - bundle_file - pending_commit - pending_bundle_commit - not_failsafe - not_valid - sys_file - secure - nosignature - public_write - public_read fileBlocksAlloc: Allocated blocks

Table 41: AT+fileGetFileList

11.4.2 File operations

In this section, the file operation commands are described.

Users shall only have one active AT+fileOpen at a time. Any AT+fileOpen shall be finalized by a AT+fileClose to ensure data integrity after filling the file using one or multiple AT+writeFile commands.

Request	Response
AT+fileOpen=[fileName],[options],[fileSize]	+fileOpen:[fileID],[secureToken] OK or error
Arguments: fileName: full file path (max 180 chars) options: - READ - Read a file (no bit mask) - WRITE - Open for writing (optionally bitmask with CREATE) - CREATE - Create a new file (optionally bitmask with WRITE or OVERWRITE) - OVERWRITE - Open an existing file (optionally bitmask with CREATE) - CREATE_FAILSAFE - CREATE_SECURE - CREATE_NOSIGNATURE (for secure files only) - CREATE_STATIC_TOKEN (for secure files only) - CREATE_VENDOR_TOKEN (for secure files only) - CREATE_PUBLIC_WRITE (for secure files only) - CREATE_PUBLIC_READ (for secure files only) fileSize: Max file size in bytes (mandatory for CREATE option)	

Table 42: AT+fileOpen

Request	Response
AT+fileClose=[fileID],[certificateFileName],[signature]	OK or error
Arguments: fileID: ID assigned from `AT+fileOpen` certificateFileName: Full path to certificate (optional) signature: The signature is SHA1 (optional)	

Table 43: AT+fileClose

Request	Response
AT+fileDel=[fileName],[secureToken]	OK or error
Arguments: fileName: Full path to file secureToken: Token assigned from `AT+fileOpen` (optional)	

Table 44: AT+fileDel

Request	Response
AT+fileGetInfo=[fileName],[secureToken]	+FileGetInfo:[Flags],[FileSize], [Allocated-Size],[Tokens],[storageSize], [WriteCounter] OK or error
Arguments: fileName: Full path to file secureToken: Token assigned from `AT+fileOpen` (optional)	

Table 45: AT+fileGetInfo

Request	Response
AT+fileRead=[fileID],[offset],[format],[length]	+FileRead:[format], [numberOfReadBytes], [data] OK or error
Arguments: fileID: ID assigned from `AT+fileOpen` offset: Offset to specific read block format: 0=binary, 1=Base64 length: Number of bytes to read	

Table 46: AT+fileRead

Request	Response
AT+fileWrite=[fileID],[offset],[format],[length],[data]	+FileWrite:[numberOfReadBytes] OK or error
Arguments: fileID: ID assigned from `AT+fileOpen` offset: Offset to specific block format: 0=binary, 1=Base64 length: Number of bytes to write (max 1460) data	

Table 47: AT+fileWrite

The module allocates memory for data read/write depending on the length field of the command. If not enough memory can be allocated an error is returned. For large files the user is required to perform fragmentation. We recommend writing chunks of up to 1024 byte per AT+fileWrite command and increasing the offset parameter with each subsequent AT+fileWrite command accordingly.

11.5 SNTP client

Cordelia-I implements an on-board SNTP client with configurable server addresses. Address of one SNTP server can be stored in the non-volatile memory. The module automatically updates the time when required (When creating any TLS connection either for device enrolment, MQTT or FOTA update).

Request	Response
AT+netAPPGet=sntp_client,server_addres	+netAPPGet:[server_address] OK or error

Table 48: SNTP get

Request	Response
AT+netAPPSet=sntp_client,server_address,[value1]	OK or error
Arguments: Option: server address (IP address or URL)	

Table 49: SNTP set

11.6 Cloud Connectivity

The Cordelia-I implements the MQTT client that creates a secure connection to the cloud for date exchange. MQTT is a machine-to-machine (M2M) connectivity protocol based on a publish/subscribe transport mechanism. Features such as light-weight, low network bandwidth and scalability make it ideal for low-power, low-bandwidth IoT applications. An MQTT network consists of a broker connected to one or more clients. Clients can each subscribe to several topics or publish any topic. The broker, on the other hand, is responsible for receiving a published topic and pushing it to all the clients having subscribed to that particular topic.

Cordelia-I offers AT commands to connect to a MQTT broker, subscribe to topics and publish data to topics. The following section describes these commands. A detailed description of individual parameters required to configure the AT command interface can be found in 6.2.3.

Request	Response
AT+iotconnect	OK or error

Table 50: AT+iotconnect

Request	Response
AT+iotdisconnect	OK or error

Table 51: AT+iotdisconnect

Request	Response
AT+iotpublish=[index],[messageString]	OK or error
Arguments: - index: Topic index (0 - 3) - message: payload (max 1011 bytes long, this can be either a plain-text string between quotation marks, or a base64 encoded string)	

Table 52: AT+iotpublish

11.7 Enrolment via QuarkLink

In order to perform the device provisioning in to the cloud end-point using the QuarkLink™ platform, a single AT command is required. This command triggers the enrolment process using the pre-configured instance of the QuarkLink™ platform. For details of the enrolment process refer to section 6.7.

Request	Response
AT+iotenrol	OK or error

Table 53: AT+iotenrol

11.8 RF test commands

Cordelia-I supports the following test commands to perform radio transmit power tests.

The module need to be configured as a STA and disconnected from any AP to perform RF tests.

11.8.1 Stop a running

This command can be used to stop any running RF tests described below.

Request	Response
AT+cordelia=0	OK or error

Table 54: AT+cordelia=0

11.8.2 Continuous transmission

This command can be used to configure the module to transmit continuously on a specific channel.

Request	Response
AT+cordelia=1,[channelIndex]	OK or error
Arguments: - channel index: 1 - 13	

Table 55: AT+cordelia=1

STOP

These commands configures the module to transmit continuously with maximum transmit power. This test mode is intended exclusively for use in the laboratory to perform RF tests.

11.8.3 Receive mode

This command can be used to configure the module in to receive mode on a specific channel.

Request	Response
AT+cordelia=2,[channelIndex]	OK or error
Arguments: - channel index: 1 - 13	

Table 56: AT+cordelia=2

11.8.4 Get Receive statistics

When in receive mode, the module collects statistics such as receive sensitivity and packet error rate. This command can be can be used to retrieve these statistics.

Request	Response
AT+cordelia=3	OK or error
Arguments: None	

Table 57: AT+cordelia=3

This command returns a comman separted value string containing the following values,

1. Sum of the packets that been received OK.
2. Sum of the packets that been dropped due to FCS error.
3. Sum of the packets that been received but filtered out by one of the HW filters.
4. Average RSSI for all valid data packets received.
5. Average RSSI for all valid management packets received.
6. Rate histogram for all valid packets received (20 values).
7. RSSI histogram from -40 until -87 (6 values. All below and above RSSI will appear in the first and last cells).
8. The time stamp started collecting the statistics in uSec.
9. The time stamp called the get statistics command.

Stop the RX test before retrieving the statistics.

11.9 Events

The host can receive an indication of specific states through events or errors. Asynchronous events can be sent to the host at any given time with an indication of specific states and specific data for each event.

11.9.1 Startup event

The startup event is output by the Cordelia-I when the AT command application has started.

Event:	
+eventstartup:[article number],[chipID],[MAC],[FW version]	
article number	Article number of the radio module
chipID	Chip ID
MAC	MAC address of the radio module
FW version	Firmware version as string

Table 58: +eventstartup event

11.9.2 General events

The general event may be received in relation to general device operation.

Event:	
+eventgeneral:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
reset_request	value1:Code
	value2:Software module - other - wlan - netcfg - netapp - security
error	value1:Code
	value2:Software module - other - wlan - netcfg - netapp - security

Table 59: +eventgeneral event

11.9.3 WLAN events

The WLAN event may be received in relation to a WLAN connection.

Event:	
+eventwlan:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
connect	value1: SSID
	value2: BSSID
disconnect	value1: SSID
	value2: BSSID
	value3: Reason, see chapter 31.2
sta_added	value1: MAC
sta_removed	value1: MAC

Table 60: +eventwlan event

11.9.4 Socket events

The socket event may be received in relation to socket operation.

Event:	
+eventsock:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
tx_failed	value1: SD
	value2: Status
async_event	value1: SD
	value2: Type - ssl_accept - rx_frag_too_big - other_side_close_ssl - connected_secured - wrong_root_ca
	value3: Error value

Table 61: +eventsock event

11.9.5 MQTT events

The MQTT event may be received in relation to one of the MQTT operations performed by the module.

Event:	
+eventmqtt:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
info	value1=Status message
	value2=Status code
error	value1=Status message
	value2=Status code
recv	value1=Topic
	value2=QoS type 0-2
	value3=Data between quotation marks

Table 62: +eventmqtt event

11.9.6 Fatal error events

The fatal error event may be received in case of device malfunction.

Event:	
+eventfatalerror:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
device_abort	value1: Code
	value2: Value
driver_abort	
sync_loss	
no_cmd_ack	value1: Code
cmd_timeout	value1: Code

Table 63: +eventfatalerror event

11.9.7 IoT events

The IoT event may be received in relation to one of the IoT operations performed by the module.

Event:	
+eventiot:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
info	value1=Status message
	value2=Status code
error	value1=Status message
	value2=Status code

Table 64: +eventiot event

11.9.8 OTA events

The OTA event may be received in relation to one of the OTA operations performed by the module.

Event:	
+eventota:[ID],[value1],. . . ,[valueX]	
ID	**[value1],. . . ,[valueX]**
info	value1=Status message
	value2=Status code
error	value1=Status message
	value2=Status code

Table 65: +eventota event

12 Provisioning

To enable easy provisioning when integrated into an embedded system with limited HMI capabilities, the Cordelia-I offers a provisioning mode. In this mode, the module acts as an AP and allows external devices with appropriate credentials to connect and access the on-board HTTP server. The user can conveniently browse the settings web page and configure the module using any web browser.

The web pages for provisioning require JavaScript.

Before the provisioning mode can be used, the user must ensure that no active WiFi connection is configured. A Cordelia-I in factory default state has no active WiFi connection.
This can be done by changing to AT command mode (you have to set both `APP_MODE` pins to LOW, and reset the module) and calling the command `AT+wlandisconnect`. Without this step, the provisioning mode won't start, and the access point won't be created!

After you have ensured that no WiFi connection is active, you can start the provisioning mode by setting `APP_MODE_1` to HIGH, and `APP_MODE_0` to LOW, and resetting the module. If the startup was successful, you should receive an event on the serial port, stating that the provisioning mode has started.

12.1 Start in provisioning mode

The application mode pins *APP_MODE_0* and *APP_MODE_1* can be used to define the application mode, as described in chapter `6.4.1`. To boot the module in the provisioning mode, apply a LOW signal to the *APP_MODE_0* pin, a HIGH signal to the *APP_MODE_1* pin and restart the module.

When the provisioning mode has been started successfully, the LED at *STATUS_IND_1* flashes with an interval of 1 s. The module has created an access point with an SSID "cordelia_" followed by the MAC of the module (example "cordelia_CAFFEE123456"). Now any WiFi enabled device can connect to the access point using WPA2 security and the key "cordeliawlan".

12.2 Add WLAN profile

On the device connected to the Cordelia-I AP, open the website "cordelia.net" in a browser.

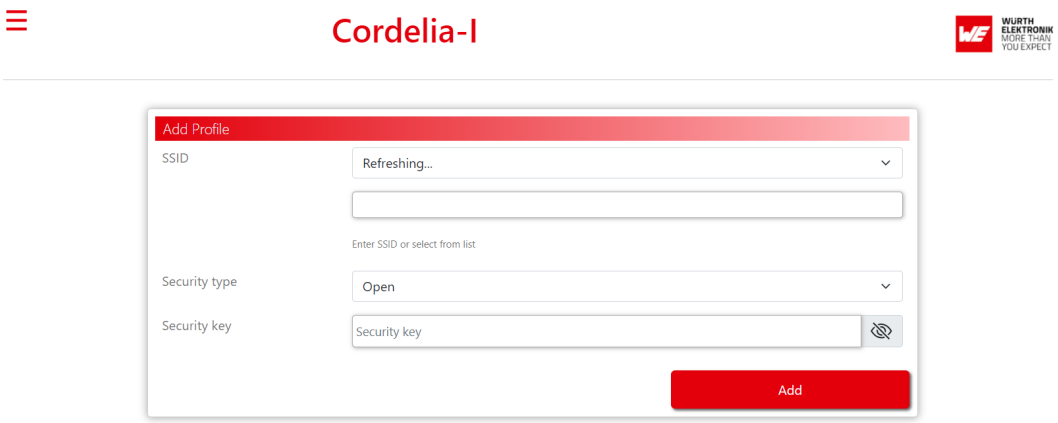


Figure 26: Provisioning main page

To save a WLAN profile in the module, select the SSID from the dropdown menu or enter the same manually in the text field. Check the correct security type, enter the key if necessary and click on the "Add" button. A pop-up appears confirming the addition of the profile.
In the default settings, the parameter "WLAN policy connection" (see chapter 11.3.5) is set to "auto|fast", meaning that the device automatically tries to connect to the access points defined in the module's profiles. Thus, after adding the profile to the module, a restart has to be performed. This can be done by sending the following command

```
AT+reboot
```

or pressing the reset button.

Please make sure that the application mode pins *APP_MODE_0* and *APP_MODE_1* are set correctly when restarting the device.

After restarting in AT command mode, the module automatically connects to the pre-defined AP.

```
+eventwlan:connect,Cordelia-Pruefrouter,0x0:0x25:0x9c:0xcf:0x85:0xf0
+eventnetapp:ipv4_acquired,192.168.1.101,192.168.1.50,192.168.1.50
```

12.3 Read and Set user settings

The provisioning pages on the Cordelia-I offer the possibility read as well as set the user settings of the Cordelia-I module. To read the value of the user setting,

- On the home page click on the "User setting" option on the menu bar.
- Click on the "GET" tab.
- From the first drop down, select the "Category" of the user setting.
- From the second drop down, select the "Setting" to read.

- Click on the "GET" button.
- The value is displayed as text.

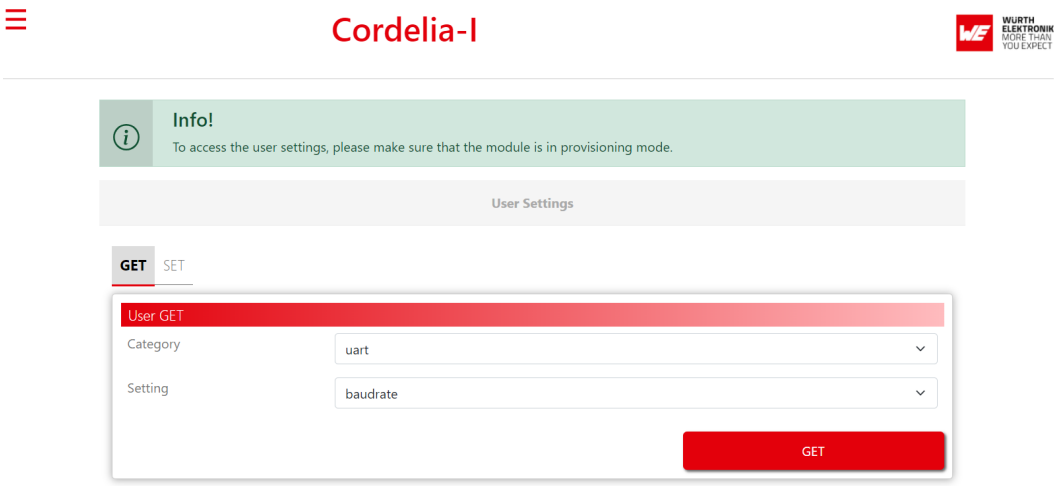


Figure 27: GET User setting

To set the value of a user setting,

- On the home page click on the "User setting" option on the menu bar.
- Click on the "SET" tab.
- From the first drop down, select the "Category" of the user setting.
- From the second drop down, select the "Setting" to read.
- Click on the "SET" button.
- The value is displayed as text.

Figure 28: SET User setting

12.4 Upload files

The provisioning pages on the Cordelia-I offer the possibility to upload files such as certificates to the the on-board file system. In order to upload a file, the following steps must be performed:

- On the home page click on the "File upload" option on the menu bar.
- Clicking the "Choose file" button opens the file browser on the device.
- Browse and select the file to be uploaded.
- Click on "Upload file" to save the file to the Cordelia-I file system.

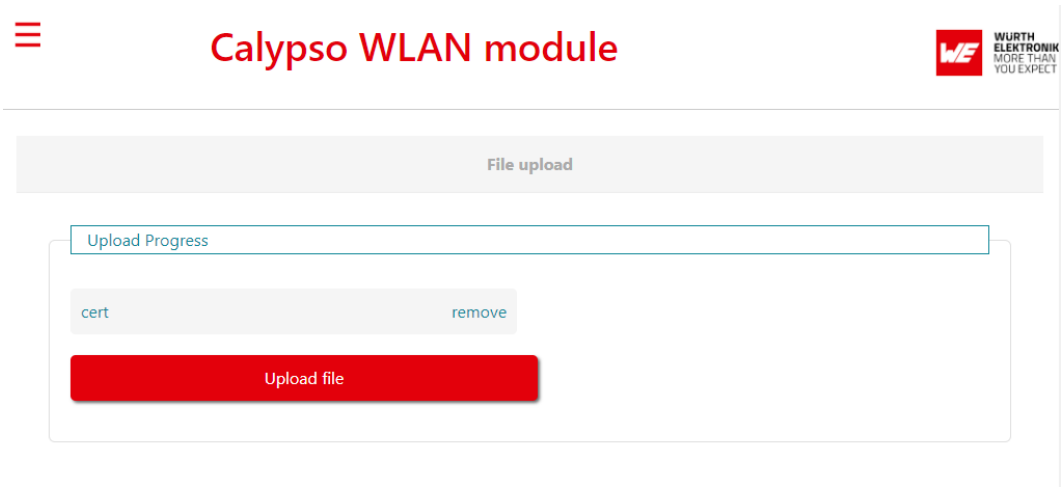


Figure 29: File upload

13 Transparent mode

The Cordelia-I radio module supports the so-called "Transparent Mode" mode of operation, which when chosen instead of the "AT commands mode", allows applications to utilize the module as a UART-to-Cloud bridge.

In transparent mode, the Cordelia-I automatically connects to a pre-configured WiFi access point and opens an MQTT connection for communication with a preconfigured remote end-point (MQTT broker). Afterwards, the Cordelia-I acts as a transparent bridge between the UART and the created cloud connection. This means that all data sent to the Cordelia-I via UART is forwarded to the MQTT-socket and all data received on the MQTT-socket is output on the UART.

The transparent mode only realizes the payload communication on exactly one MQTT connection. Due to it's intended usecase it will not provide any evens (such as a disconnect or server timeout) on the transparent interface towards the host. This interface will only send the received data from the MQTT connection to the host. The AT command mode is needed for any actions and configuration changes that are intended with the Cordelia-I.

13.1 Work flow

When booting the Cordelia-I radio module, the voltage level of the pins *APP_MODE_0* and *APP_MODE_1* is checked to detect the mode of operation. In case both pins have a high level, the Cordelia-I starts in transparent mode.

In transparent mode, the Cordelia-I first tries to connect to a WiFi access point. In case this fails due to the absence of the configured access point or wrong access point credentials, it retries until a connection is established.

On success, the pin *STATUS_IND_0* turns high and the Cordelia-I tries to open a MQTT-socket. In case of failure, it retries until a MQTT-socket has been opened.

As soon as a socket has been opened and the connection has been established successfully, the pin *STATUS_IND_1* turns high and the UART of the Cordelia-I is switched on. All UART data is forwarded to the socket and vice versa. In this state the pin */RTS* of the Cordelia-I indicates when the radio module is ready to receive data via UART. If the UART flow control is disabled, the pin stays active (LOW) as long as a socket is available. In case the UART flow control is enabled, the */RTS* pin stays active (LOW) as long as a socket is available and the UART is ready to receive more data.

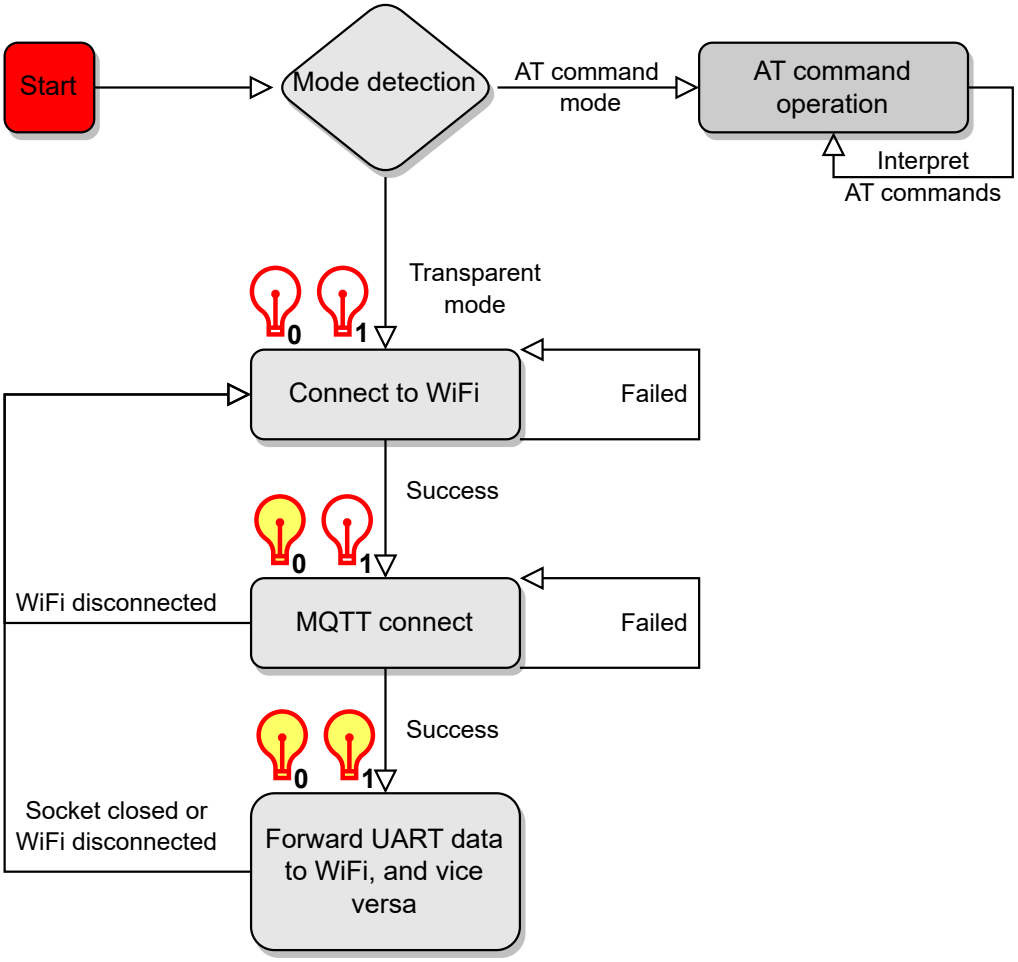


Figure 30: Flow chart - transparent mode

Before starting the module in transparent mode, various settings have to be defined in order to ensure successful WiFi connection and MQTT-socket set-up. For details regarding the required steps, please refer to the following sections.

13.2 WiFi connection set-up

To successfully set up a connection to a WiFi access point, the user has to enter the network credentials such as SSID, password and security mode into the Cordelia-I in advance. There are two ways of doing so:

1. Use the AT-commands interface to configure the Cordelia-I via UART (See section 11.3.6).
2. Use the web interface to configure the module (See section 12.2).

13.3 MQTT connection set-up

In case the Cordelia-I radio module has successfully set up a connection to an access point in transparent mode, the pin *STATUS_IND_0* turns high. The next step is to open a MQTT-socket.

To enable the successful set-up of a socket, various settings must be defined in advance. There are two ways of doing so:

1. Use the AT-commands interface to enter the MQTT settings via UART (See section 11.2).
2. Use the web interface to enter the MQTT settings (See section 12.3).
3. Use the QuarkLink server to enrol and securely provision the device (See section 6.7).

13.4 Transparent UART set-up

Finally, it is essential to set up the UART interface in order to enable data transmission and reception in the transparent mode. Ensure that the parameters are set appropriately using one of the following methods,

1. Use the AT-commands interface to enter the UART settings (See section 11.2).
2. Use the web interface to enter the UART settings (See section 12.3).

To prevent loss of data, users of the transparent mode shall enable the Flow Control and implement the according actions on their host side.

The following parameters need to be set:

- **baudrate**:is set to 115200 bit/s by default can take a value up to 3 Mbit/s. Use of flow control is recommended for rates higher than 921600 bit/s.
- **parity**: is set to "none" by default.
- **flowcontrol**: is not set by default, but should be enabled for transparent mode and any baudrate.
- **transparent_trigger**: is a flag that determines the trigger to start wireless data transmission in the transparent mode. The trigger can be receipt one or two specific end-of-transmission characters or a timeout. The default value us "2etx|timer" (trigger on 2 end-of-transmission characters and timeout)
- **transparent_timeout**: is the time after which a transmission is triggered. Default value is 20 ms.
- **transparet_etx**: The set of characters that trigger a transmission in the transparent mode. The default value is 0x0D0A ("\r\n")

13.5 Data exchange

In order to start the data exchange, boot the Cordelia-I radio module in transparent mode by applying the voltage level high on the pins *APP_MODE_0* and *APP_MODE_1*.

In transparent mode, the Cordelia-I first tries to connect to a WiFi access point configured in section 13.2. In case this fails due to the absence of the configured access point or wrong access point credentials, it retries until a connection is established.

On success, the pin *STATUS_IND_0* turns high and the Cordelia-I tries to open a MQTT-socket with the configuration set in section 13.3. In case of failure, it retries until a MQTT-socket has been opened.

As soon as a socket has been opened and the connection has been established successfully, the pin *STATUS_IND_1* turns high and the UART of the Cordelia-I is switched on. All UART data is forwarded to the socket and vice versa. In this state the pin */RTS* of the Cordelia-I indicates when the radio module is ready to receive data via UART. If the UART flow control is disabled, the pin stays active (LOW) as long as a socket is available. In case the UART flow control is enabled, the */RTS* pin stays active (LOW) as long as a socket is available and the UART is ready to receive more data.

14 Use cases and examples

This section presents a variety of practical use cases and examples to help you effectively utilize the Cordelia-I module in different scenarios. We recommend following the examples in the order presented, as they gradually increase in complexity while reinforcing core concepts. The first set of examples focuses on establishing basic non-secure and secure MQTT connections to the publicly available test.mosquitto.org broker, using both manual AT commands and WE UART Terminal software. Next, we demonstrate how to connect the module to a QuarkLink instance, providing insights into two-way authentication. Following this, you will learn how to set up your own non-secure and secure MQTT brokers on a Raspberry Pi, enabling local device communication with various security measures. Finally, we conclude with an example of connecting two Cordelia-I modules in transparent mode, showcasing the flexibility of the MQTT protocol. Each example is designed to build upon the last, ensuring a comprehensive understanding of the module's capabilities.

For a smoother experience, we also recommend reading the Quick Start Guide in Section 7.5.3 before beginning with these examples, as it provides essential setup information and an introduction to key concepts.

Let's dive in and explore the capabilities of your Cordelia-I module together!

14.1 Connecting the Cordelia-I module to test.mosquitto.org with a basic non-secure MQTT connection using manual AT commands

In this tutorial, we'll guide you through connecting the Cordelia-I module to test.mosquitto.org, a publicly available Eclipse Mosquitto MQTT server/broker widely used for general testing of MQTT applications. We'll initiate a basic, non-secure MQTT connection, aiming to get an MQTT connection up and running and send some data in the simplest way possible. **Please note that this communication is not secure, so avoid sending any sensitive data using this method!** We'll be sending AT commands to the Cordelia-I module manually.

We'll use the serial terminal program HTerm to send commands and view responses from the module. Let's get started with setting up the module and establishing a wireless communication link!

1. Take your Cordelia-I evaluation board and connect it to your PC with a USB cable. By default, the device starts up in AT command mode, and it uses a virtual COM port (115200 baud, 8N1) to communicate with the PC.
2. Open a serial terminal program. In this tutorial, we are using HTerm. Select the COM port that your Cordelia-I evaluation board is connected to, and open the COM port to enable your computer to communicate with the module. Set up the parameters of the COM port to 115200 baud, 8 data bits, no parity bit, and 1 stop bit. In the transmit window, ensure that each command you send is terminated with a CR-LF sequence; otherwise, the Cordelia-I module might have problems interpreting the command.
3. Press the Reset button on your evaluation board. If the COM port is set up correctly, you should see a startup message that looks like this:

```
+eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
```

4. First, connect your Cordelia-I to a WLAN network by sending the following command (replace YOUR_SSID with your WLAN SSID and YOUR_PASSWORD with your WLAN password):

   ```
   AT+wlanconnect=YOUR_SSID,,WPA_WPA2,YOUR_PASSWORD,,,
   ```

5. If the WLAN connection was successful, the status LED on your evaluation board should light up, and you should see the following response events on the serial port:

   ```
   +eventwlan:connect,YOUR_SSID,0x34:0x60:0xf9:0x21:0x79:0x61
   +eventnetapp:ipv4_acquired,192.168.1.122,192.168.1.1,192.168.1.1
   ```

6. Now, set up the basic user settings required to initiate a basic MQTT connection. First, set up the iotHubEndpoint, which is the URL or IP address of the MQTT broker to connect to. In this case, it is "test.mosquitto.org". To set the iotHubEndpoint parameter, go to the serial terminal program and send the following command:

   ```
   AT+set=MQTT,iotHubEndpoint,"test.mosquitto.org"
   ```

   If the setup was successful, the Cordelia-I module responds with an "OK" response.

7. Then, set up the iotHubPort, which is the port where the server is listening. Generally, this is port 1883 for non-secure communication, and port 8883 for secure communication. For a complete list of available ports, you can open test.mosquitto.org in your web browser. Here, we set up a basic, non-secure communication using port 1883. To set the iotHubPort parameter, go to the serial terminal program and send the following command:

   ```
   AT+set=MQTT,iotHubPort,1883
   ```

   Note that this is a numeric parameter, so quotation marks are not needed. If the setup was successful, the Cordelia-I module responds with an "OK" response.

8. Now, set the clientId. The clientId is a unique identifier distinguishing each MQTT client connecting to a broker and enables the broker to keep track of the client's current state. To set the clientId parameter, go to the serial terminal program and send the following command:

   ```
   AT+set=MQTT,clientId,"Cordelia1234"
   ```

   This will set the clientId to "Cordelia1234". However, it is recommended to use a unique clientId, as trying to connect a second module with the same clientId will be unsuccessful. If the setup was successful, the Cordelia-I module responds with an "OK" response.

9. Next, set up the MQTT connection flags. You can read about the available options in the detailed AT command description section of this user manual. In this case, as we are using a URL as an endpoint, not an IP address, we need to set the url flag. To set the MQTT connection flags, go to the serial terminal program and send the following command:

   ```
   AT+set=MQTT,flags,"url"
   ```

   If the setup was successful, the Cordelia-I module responds with an "OK" response.

10. To test if the connection was successful, subscribe to an MQTT topic by setting the subtopic0 parameter to "apple". This means that after connection, the Cordelia-I module will subscribe to the "apple" topic and listen for incoming messages. Although you can set up to four subtopics (index 0 to 3), one is sufficient for this tutorial. To set subtopic0 to "apple", go to the serial terminal program and send the following command:

    ```
    AT+set=SUBTOPIC0,name,"apple"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

11. Set up the topic you want to publish to. In this case, we want to publish to the "apple" topic, so that everything published will echo back to the module, allowing verification. To set pubtopic0 to "apple", go to the serial terminal program and send the following command:

    ```
    AT+set=PUBTOPIC0,name,"apple"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

12. All parameters required for a basic MQTT connection are now set up, meaning we are ready to initiate a connection. To do this, go to the serial terminal program and send the following command:

    ```
    AT+iotconnect
    ```

    If the connection was successful, the Cordelia-I module responds with an "OK" response, and after a few seconds, you should receive a "CONNACK" event, indicating that the connection to the MQTT broker was set up successfully and you are ready to send and receive messages.

13. Finally, let's publish some messages. Go to the serial terminal program and send the following command:

    ```
    AT+iotpublish=0,"Hello␣Cordelia!"
    ```

    Since we set the pubtopic with index 0 to "apple", and are also subscribed to the topic "apple", we should receive the message back. If the publishing was successful, you will first receive an "OK" response from the Cordelia-I module. Then, since we are subscribed to the "apple" topic, a +eventmqtt:recv event will appear containing the message you just sent:

    ```
    +eventmqtt:recv,apple,qos0,"Hello␣Cordelia!"
    ```

14. Congratulations! You successfully connected your Cordelia-I module to test.mosquitto.org using a basic MQTT connection. Feel free to play around, try other parameters, and send any kind of data you want.

14.2 Connecting the Cordelia-I module to test.mosquitto.org with a secure MQTT connection using manual AT commands

In this tutorial, we'll guide you through connecting the Cordelia-I module to test.mosquitto.org, a publicly available Eclipse Mosquitto MQTT server/broker widely used for general testing of MQTT applications. We'll initiate a secure, encrypted, unauthenticated MQTT connection using the Root CA available on the test.mosquitto.org website. **Please note that this example**

is for demonstration purposes only, and you should not use the test.mosquitto.org broker for any sensitive data transfer. This tutorial is similar to the one where we connect to test.mosquitto.org in a non-secure way, but here we're taking extra steps to ensure the connection is secure. We'll be sending AT commands to the Cordelia-I module manually.

In this tutorial, we're using one-way authentication. One-way authentication is the simplest form of authentication used in TLS. In one-way authentication, the server presents its digital certificate to the client. The client verifies the certificate to ensure it is valid and issued by a trusted CA. Once the certificate is confirmed, the client can establish a secure connection with the server. One-way authentication is sufficient when the client does not need to be authenticated or when the client's identity is not critical.

We'll use the serial terminal program HTerm to send commands and view responses from the module. Let's get started with setting up the module and establishing a wireless communication link!

1. Take your Cordelia-I evaluation board and connect it to your PC with a USB cable. By default, the device starts up in AT command mode, and it uses a virtual COM port (115200 baud, 8N1) to communicate with the PC.

2. Open a serial terminal program. In this tutorial, we are using HTerm. Select the COM port that your Cordelia-I evaluation board is connected to, and open the COM port to enable your computer to communicate with the module. Set the parameters of the COM port to 115200 baud, 8 data bits, no parity bit, and 1 stop bit. In the transmit window, ensure that each command you send is terminated with a CR-LF sequence; otherwise, the Cordelia-I module might have problems interpreting the command.

3. Press the Reset button on your evaluation board. If the COM port is set up correctly, you should see a startup message that looks like this:

   ```
   +eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
   ```

4. First, connect your Cordelia-I to a WLAN network. To do this, send the following command (replace `YOUR_SSID` with your WLAN SSID and `YOUR_PASSWORD` with your WLAN password):

   ```
   AT+wlanconnect=YOUR_SSID,,WPA\_WPA2,YOUR_PASSWORD,,,
   ```

5. If the WLAN connection is successful, the status LED on your evaluation board should light up, and you should see the following response events on the serial port:

   ```
   +eventwlan:connect,YOUR_SSID,0x34:0x60:0xf9:0x21:0x79:0x61
   +eventnetapp:ipv4_acquired,192.168.1.122,192.168.1.1,192.168.1.1
   ```

6. Set up the basic user settings required to initiate a secure MQTT connection. First, configure the `iotHubEndpoint`, which is the URL or IP address of the MQTT broker. In this case, it is `"test.mosquitto.org"`. To set the `ClientId` parameter, go to the serial terminal program and send the command:

   ```
   AT+set=MQTT,iotHubEndpoint,"test.mosquitto.org"
   ```

   If the setup is successful, the Cordelia-I module responds with an "OK" response.

7. Set up the `iotHubPort`, which is the port where the server is listening. Generally, this is port 1883 for non-secure communication and port 8883 for secure communication. For a secure, encrypted, unauthenticated MQTT connection, use `"8883"`. To set the `iotHubPort` parameter, send the command:

```
AT+set=MQTT,iotHubPort,8883
```

   If successful, the Cordelia-I module responds with an "OK" response.

8. Set the `clientId`. The `clientId` is a unique identifier that distinguishes each MQTT client connecting to a broker. To set the `clientId` parameter, send the command:

```
AT+set=MQTT,clientId,"Cordelia1234"
```

   If the setup is successful, the Cordelia-I module responds with an "OK" response.

9. Set up the MQTT connection flags. Use a URL as an endpoint, set the `url` and `sec` flags, and enable the `whitelist_rootca` flag to whitelist the provided root CA. Send the command:

```
AT+set=MQTT,flags,"url|sec|whitelist_rootca"
```

   If successful, the Cordelia-I module responds with an "OK" response.

10. Upload the Root CA of the test.mosquitto.org broker to the Cordelia-I module. Use the `FileOpen`, `FileWrite`, and `FileClose` commands. To upload the Root CA as `"mosquitto.org.pem"`, the upload process should look like this:

```
AT+FileOpen=mosquitto.org.pem,WRITE|CREATE,1452

+fileopen:312865874,0
OK

AT+FileWrite=312865874,0,0,512,-----BEGIN CERTIFICATE-----
MIIEAzCCAuugAwIBAgIUBY1hlCGvdj4NhBXkZ/uLUZNILAwwDQYJKoZIhvcNAQEL
BQAwgZAxCzAJBgNVBAYTAkdCMRcwFQYDVQQIDA5Vbmlod2luZ2RvbTEOMAwG
A1UEBwwFRGVyYnkxEjAQBgNVBAoMCU1vc3F1aXRObzELMAkGA1UECwwCQ0ExFjAU
BgNVBAMMDW1vc3F1aXRObyBvcmcxHzAdBgkqhkiG9w0BCQEWEHJvZ2VyQGF0Y2hv
by5vcmcwHhcNMjAwNjA5MTEwNjM5WhcNMzAwNjA3MTEwNjM5WjCBkDELMAkGA1UE
BhMCR0IxFzAVBgNVBAgMDlVuaXRlZCBLaW5nZG9tMQ4wDAYDVQQHDAVEZXJieTES
MBAGA1UECgwJTW9zcXVpdHRvMQswCQYDVQQLDAJDQTEWMBQGA1UEAwwNbW9zcXVp
dHRvLm9yZzEfMB0GCSqGSIb3DQEJA

+filewrite:512
OK

AT+FileWrite=312865874,512,0,512,RYQcm9nZXJAYXRjaG9vLm9yZzCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCCAQoCggEBAME0HKmIzfTOwkKLT3THHe+ObdizamPg
UZmD64Tf3zJdNeYGYn4CEXbyP6fy3tWc8S2boW6dzrH8SdFf9uo320GJA9B7U1FW
Te3xda/Lm3JFfaHjkWw7jBwcauQZjpGINHapHRlpiCZsquAthOgxW9SgDgYlGzEA
s06pkEFiMw+qDfLo/sxFKB6vQlFekMeCymjLCbNwPJyqyhFmPWwio/PDMruBTzPH
3cioBnrJWKXc3OjXdLGFJOfj7pP0j/dr2LH72eSvv3PQQFl90CZPFhrCUcRHSSxo
E6yjGOdnz7f6PveLIB574kQORwt8ePn0yidrTC1ictikED3nHYhMUOUCAwEAAaNT
MFEwHQYDVR0OBBYEFPVV6xBUFPiGKDyo5V3+Hbh4N9YSMB8GA1UdIwQYMBaAFPVV
6xBUFPiGKDyo5V3+Hbh4N

+filewrite:512
OK
```

```
AT+FileWrite=312865874,1024,0,428,9YSMA8GA1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQEL
BQADggEBAGa9kS21N70ThM6/Hj9D7mbVxKLBjVWe2TPsGfbl3rEDfZ+OKRZ2j6AC
6r7jb4TZO3dzF2p6dgbrlU71Y/4K0TdzIjRj3cQ3KSm41JvUQ0hZ/c04iGDg/xWf
+pp58nfPAYwuerruPNWmlStWAXf0UTqRtg4hQDWBuUFDJTuWuuBvEXudz74eh/wK
sMwfu1HFvjy5Z0iMDU8PUDepjVolOCue9ashlS4EB5IECdSR2TItnAIiIwimx839
LdUdRudafMu5T5Xma182OC0/u/xRlEm+tvKGGmfFcN0piqVl8OrSPBgIlb+1IKJE
m/XriWr/Cq4h/JfB7NTsezVslgkBaoU=
-----END CERTIFICATE-----

+filewrite:428
OK

AT+FileClose=312865874,,
OK
```

11. Set the `rootCAPath` parameter so that the Cordelia-I module knows which Root CA to use for the connection. Send the command:

    ```
    AT+set=MQTT,rootCAPath,"mosquitto.org.pem"
    ```

    If successful, the Cordelia-I module responds with an "OK" response.

12. Test the connection by subscribing and publishing to some MQTT topics. Set the `subtopic0` parameter to `"apple"` by sending the command:

    ```
    AT+set=SUBTOPIC0,name,"apple"
    ```

    If successful, the Cordelia-I module responds with an "OK" response.

13. To publish to a topic, set the `pubtopic0` parameter to `"apple"` by sending the command:

    ```
    AT+set=PUBTOPIC0,name,"apple"
    ```

    If successful, the Cordelia-I module responds with an "OK" response.

14. All parameters required for a basic MQTT connection are set. Initiate a connection by sending the command:

    ```
    AT+iotconnect
    ```

    If successful, the Cordelia-I module responds with an "OK" response, and after a few seconds, you should receive a `"CONNACK"` event, indicating that the connection to the MQTT broker was successful.

15. Try publishing some messages by sending the command:

    ```
    AT+iotpublish=0,"Hello␣Cordelia!"
    ```

    If successful, you receive an `OK` response and a `recv` event containing the message you just sent:

    ```
    +eventmqtt:recv,apple,qos0,"Hello␣Cordelia!"
    ```

16. Congratulations! You successfully connected your Cordelia-I module to `test.mosquitto.org` using a secure, encrypted, unauthenticated MQTT connection.

14.3 Connecting the Cordelia-I module to test.mosquitto.org with a basic non-secure MQTT connection using WE UART Terminal

In this tutorial, we'll guide you through connecting the Cordelia-I module to test.mosquitto.org, a publicly available Eclipse Mosquitto MQTT server/broker widely used for general testing of MQTT applications. We'll initiate a basic, non-secure MQTT connection, aiming to get an MQTT connection up and running and send some data in the simplest way possible. **Please note that this communication is not secure, so avoid sending any sensitive data using this method!** We'll be sending AT commands to the Cordelia-I module using the WE UART Terminal software, which greatly simplifies the process.

We'll use the WE UART Terminal software to send commands and view responses from the module. Let's get started with setting up the module and establishing a wireless communication link!

1. Take your Cordelia-I evaluation board and connect it to your PC with a USB cable. By default, the device starts up in AT command mode, and it uses a virtual COM port (115200 baud, 8N1) to communicate with the PC.

2. Open the WE UART Terminal software. Select the COM port that your Cordelia-I evaluation board is connected to and open the COM port to enable your computer to communicate with the module. Set up the parameters of the COM port to 115200 baud, 8 data bits, no parity bit, and 1 stop bit.

3. Press the Reset button on your evaluation board. If the COM port is set up correctly, you should see a startup message that looks like this:

```
+eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
```

4. Go to the "WLAN settings" tab and replace the WLAN_SSID and WLAN_PASSWORD entry fields with your WLAN SSID and WLAN password. Then, click "Connect." If the WLAN connection was successful, the status LED on your evaluation board should light up, and you should see the following response events on the serial port:

```
+eventwlan:connect,YOUR_SSID,0x34:0x60:0xf9:0x21:0x79:0x61
+eventnetapp:ipv4_acquired,192.168.1.122,192.168.1.1,192.168.1.1
```

5. Go to the "General" tab and click the button "Set up module with dummy parameters." This will set up the module with some dummy parameters that are working out-of-the-box. Be aware that this means that a dummy value is going to be used for the MQTT connection, and even the clientId will be set up to a dummy value. If someone is already connected to test.mosquitto.org using the same clientId, the connection will be unsuccessful. The WE UART Terminal then should set up the module with all the required dummy parameters. The following commands are being sent out:

```
AT+set=CSR,country,"DE"
AT+set=CSR,state,"Bavaria"
AT+set=CSR,locality,"Munich"
AT+set=CSR,surname,"Mustermann"
AT+set=CSR,email,"info@example.com"
AT+set=CSR,organization,"Generic␣GmbH"
AT+set=CSR,unit,"Wireless␣Solutions"
```

```
AT+set=QUARKLINK,hostname,"https://generic.quarklink.io"
AT+set=QUARKLINK,port,6000
AT+set=QUARKLINK,rootCAPath,"/ql/ql_ca.pem"
AT+set=MQTT,iotHubEndpoint,"test.mosquitto.org"
AT+set=MQTT,iotHubPort,1883
AT+set=MQTT,clientCertPath,""
AT+set=MQTT,rootCAPath,"mosquitto.org.pem"
AT+set=MQTT,clientPrivateKey,""
AT+set=MQTT,clientId,"Cordelia1234"
AT+set=MQTT,flags,"url"
AT+set=MQTT,base64,0
AT+set=MQTT,willTopic,""
AT+set=MQTT,willMessage,""
AT+set=MQTT,willQos,0
AT+set=MQTT,willRetain,0
AT+set=MQTT,userName,""
AT+set=MQTT,password,""
AT+set=MQTT,keepAliveTimeOut,0
AT+set=MQTT,cleanConnect,1
AT+set=SUBTOPIC0,name,"apple"
AT+set=SUBTOPIC0,qos,0
AT+set=SUBTOPIC1,name,"banana"
AT+set=SUBTOPIC1,qos,0
AT+set=SUBTOPIC2,name,"orange"
AT+set=SUBTOPIC2,qos,0
AT+set=SUBTOPIC3,name,"kiwi"
AT+set=SUBTOPIC3,qos,0
AT+set=PUBTOPIC0,name,"apple"
AT+set=PUBTOPIC0,qos,0
AT+set=PUBTOPIC0,retain,0
AT+set=PUBTOPIC1,name,"banana"
AT+set=PUBTOPIC1,qos,0
AT+set=PUBTOPIC1,retain,0
AT+set=PUBTOPIC2,name,"orange"
AT+set=PUBTOPIC2,qos,0
AT+set=PUBTOPIC2,retain,0
AT+set=PUBTOPIC3,name,"kiwi"
AT+set=PUBTOPIC3,qos,0
AT+set=PUBTOPIC3,retain,0
AT+set=OTA,url,"https://www.we-online.com/res/wco/Cordelia"
AT+set=OTA,rootcapath,"otarootca.pem"
```

6. Go to the "IoT operations" tab and click "Connect." If the connection was successful, the Cordelia-I module responds with an "OK" response, and after a few seconds, you should receive a "CONNACK" event, indicating that the connection to the MQTT broker was set up successfully and you are ready to send and receive messages.

7. Now, let's try to actually publish some messages! Go to the "IoT operations" tab, to the "Publish data" section, and click the "Publish" button. This will publish the message "This is a test message." to the pubtopic with the index 0. Remember, because we set the pubtopic with index 0 to "apple," and we are also subscribed to the topic "apple," we should get the message back. And that is exactly what is happening! If the publishing was successful, first, you get an OK response back from the Cordelia-I module, and after a short time, because we are subscribed to the "apple" topic, you should get a receive event containing the message you just sent out:

```
+eventmqtt:recv,apple,qos0,"This␣is␣a␣test␣message."
```

8. Congratulations! You successfully connected your Cordelia-I module to test.mosquitto.org using a basic MQTT connection with the WE UART Terminal software. Feel free to play around, try other parameters, and send any kind of data you want.

14.4 Connecting a Cordelia-I module to a QuarkLink instance with a secure MQTT connection using manual AT commands

In this tutorial, we'll guide you through connecting the Cordelia-I module to a QuarkLink instance. We'll initiate a secure, encrypted, unauthenticated MQTT connection using the certificates provided by the QuarkLink portal that we're enrolling to. This tutorial is an advanced version of the Quick Start Guide example. We'll achieve the same result as in the Quick Start Guide, but we'll delve deeper to understand what's happening in the background.

In this tutorial, we're using two-way authentication. Two-way authentication, or mutual authentication (mTLS), is a more secure form of authentication used in TLS. In two-way authentication, both the client and the server authenticate each other. The client presents its digital certificate to the server, and the server verifies the certificate to ensure it is valid and issued by a trusted CA. The server also presents its digital certificate to the client, and the client verifies the certificate to ensure it is valid and issued by a trusted CA. Once both certificates are verified, the client and the server can establish a secure connection. Two-way authentication is used when both the client and the server need to be authenticated or when the client's identity is critical.

We'll use the serial terminal program HTerm to send commands and view responses from the module. Let's get started with setting up the module and establishing a wireless communication link!

1. Open up a web browser on your PC and go to `https://signup.quarklink.io/`! Here, you can register your personal QuarkLink instance for free. After registration, you'll have a platform where you can configure and maintain all of your Cordelia-I modules, with useful features like an inbuilt MQTT broker and a serial terminal emulator.

2. After successful registration, you should be able to access your personal QuarkLink instance via the link provided at the end of the registration process. This link will look something like `https://abcdefghij1k2l.quarklink.io/`.

3. Go to your personal QuarkLink instance using that link and log in with the credentials you provided during registration.

4. After logging in, you'll find yourself on the QuarkLink main dashboard. From here, we'll add our Cordelia-I module to the QuarkLink instance and enrol it, activating it for data transfer in your selected IoT Hub (by default, this is the built-in MQTT broker in your QuarkLink instance).

5. Go to the "HSM" menu item, and click the "OEMRoot" certificate, then click the "Download" button to download the certificate in PEM format. We are going to upload this file to the Cordelia-I module in a later step to ensure secure enrolment to the QuarkLink portal.

6. Take your Cordelia-I evaluation board and connect it to your PC with a USB cable. By default, the device starts up in AT command mode and uses a virtual COM port (115200 baud, 8N1) to communicate with the PC.

7. Open up a serial terminal program. In this tutorial, we are using HTerm. Select the COM port that your Cordelia-I evaluation board is connected to, and open the COM port to enable your computer to communicate with the module. Set up the parameters of the COM port to 115200 baud, 8 data bits, no parity bit, and 1 stop bit. In the transmit window, ensure that if you send a command, it is always terminated with a CR-LF sequence; otherwise, the Cordelia-I module might have problems interpreting the command.

8. Press the Reset button on your evaluation board. If the COM port is set up correctly, you should see a startup message that looks like this:

```
+eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
```

9. Before you start configuring the module, you should add it to the default batch on your QuarkLink instance. To do this, retrieve the device ID of your Cordelia-I module. You can do this by going to the serial terminal program and sending the following command: “AT+get=iot,deviceid“. The Cordelia-I module then responds with the device ID:

```
+get:7054cb2c17afc4677fcfa8f075914fd7a7383e33e070a25a4af44e3045be21f2
```

10. Go to your QuarkLink instance in your web browser, then select the "Devices" menu item. There, find the "Default" batch, and click on the "Edit batch" button. Here, you can add your device to your batch by copying the device ID into the text entry field and clicking the "Add" button. After adding your device, click the "Save" button to save your batch. Now your device has been added, but it is not active yet. For that, you have to enrol your Cordelia-I module, as described in the upcoming tutorial steps.

11. Now, go back to the serial terminal program for your Cordelia-I evaluation board. First, you should connect your Cordelia-I to a WLAN network. For this, send the following command (replace YOUR_SSID with your WLAN SSID and YOUR_PASSWORD with your WLAN password):

```
AT+wlanconnect=YOUR_SSID,,WPA_WPA2,YOUR_PASSWORD,,,‘‘
```

12. If the WLAN connection was successful, the status LED on your evaluation board should light up, and you should see the following response events on the serial port:

```
+eventwlan:connect,YOUR_SSID,0x34:0x60:0xf9:0x21:0x79:0x61
+eventnetapp:ipv4_acquired,192.168.1.122,192.168.1.1,192.168.1.1
```

13. Now, we have to upload the downloaded OEMRoot certificate PEM file to the Cordelia-I module. For this, use the FileOpen, FileWrite, and FileClose commands. It is recommended to use 512-byte large batches when uploading files via AT commands. For further reference, read the description of the file commands in the AT commands section. If you want to upload the Root CA to the file path “/ql/ql_ca.pem“, the upload process should look like this (the certificate content here is a dummy value, you should use your own certificate):

```
AT+FileOpen=/ql/ql_ca.pem,WRITE|CREATE,534

+fileopen:287050753,0
OK

AT+FileWrite=287050753,0,0,512,-----BEGIN CERTIFICATE-----
MsgfergtwertwertwergtrhrtDFHGHDRgERgsfdsdfhG345GWEG54gdfsgUEAxMH
T0VNUm9vdDdfsgsdfgsdfgfgsdfgsdfgsdfgMDU0MDIxMzEwNTc0Mlo56zu57A4G
A156564jsdfjklsdfoweriopwerfiodjosdgjkvsduioprejiferkfref74QJkWX
0ac7LYAJgP0jBziHsdfgsdfgdgserzw54z63453gt54363KYLvRZHc2qFwuRhIdF
MU6LZDkoU3WjQjBAMAgergtregergwQEAwIChDgrdfhhghMBAf8EBTADAQH/MB0G
A1UdDgQWBBQJtup0YM3p7d6WfJlTxOgfhdfghdfghdghdfghPQQDAgNIADBFAiBf
F4ZzuSWVlEwq4Qvfdghdghdghtr6z56ztzetzezetzerz5z5665473u6u9Upr/Zt
sfhdfhh56546h4545456z565
----

+filewrite:512
OK

AT+FileWrite=287050753,512,0,22,-END CERTIFICATE-----

+filewrite:22
OK

AT+FileClose=287050753,,
OK
```

14. Next, we set up the basic user settings required to initiate the enrolment to the QuarkLink instance. First, set up the QuarkLink hostname. This is the link to your personal QuarkLink instance provided at the end of the registration process. This link will look something like `https://abcdefghij1k2l.quarklink.io/`. To set the QuarkLink hostname parameter, go to the serial terminal program and send the following command:

    ```
    AT+set=QUARKLINK,hostname,"https://wurth.quarklink.io"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

15. Then, set up the QuarkLink port. Generally, this is port 6000. To set the QuarkLink port parameter, go to the serial terminal program and send the following command:

    ```
    AT+set=QUARKLINK,port,6000
    ```

    Note that here we sent a number as a parameter, not a string, so we didn't use quotation marks. If the setup was successful, the Cordelia-I module responds with an "OK" response.

16. Set the QuarkLink rootCAPath to the file we uploaded three steps before. This is needed so the Cordelia-I module knows which Root CA to use for the QuarkLink enrolment. To set the QuarkLink rootCAPath parameter, go to the serial terminal program and send the following command:

    ```
    AT+set=QUARKLINK,rootCAPath,"/ql/ql_ca.pem"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

17. Now, set the CSR parameters required for the correct enrolment of the Cordelia-I module. Set up the CSR parameters by sending the following commands (customize them to fit

your profile and application). If the setup was successful, the Cordelia-I module responds with an "OK" response to each command:

```
AT+set=CSR,country,"DE"
AT+set=CSR,state,"Bavaria"
AT+set=CSR,locality,"Munich"
AT+set=CSR,surname,"Mustermann"
AT+set=CSR,email,"info@example.com"
AT+set=CSR,organization,"Generic␣GmbH"
AT+set=CSR,unit,"Wireless␣Solutions"
```

18. Initiate the enrolment process. To start the enrolment, go to the serial terminal program and send the following command:

    ```
    AT+iotenrol
    ```

    If the enrolment was successful, the Cordelia-I module responds with the following response events:

    ```
    +eventiot:info,"GET_TEMP_CSR",0
    +eventiot:info,"CREATE_TEMP_CSR",0
    +eventiot:info,"SENDING_CSR_TO_QL",0
    +eventiot:info,"CONNECTING_TO_HOST",0
    +eventiot:info,"HTTP_RESPONSE_CODE",200
    +eventiot:info,"TEMP_CERT_RECEIVED",790
    +eventiot:info,"WRITING_TEMP_CERT",0
    +eventiot:info,"TEMP_CERT_SAVED",0
    +eventiot:info,"CREATE_DEVICE_CSR",0
    +eventiot:info,"CONNECTING_QL_TLS",0
    +eventiot:info,"CONNECTING_TO_HOST",0
    +eventiot:info,"QL_RESPONSE_RECEIVED",0
    +eventiot:info,"ENROLMENT_COMPLETE",0
    ```

19. Finally, set up the MQTT connection flags. Read about the available options in the detailed AT command description section of this user manual. In this case, we are using a URL as an endpoint, not an IP address, so set the url flag in the module. Besides that, set the secure “sec“ flag and the “whitelist_rootca“ flag, which indicates that we whitelist the provided root CA, bypassing the root CA catalog verification (this is required for the QuarkLink instance). To set the MQTT connection flags, go to the serial terminal program and send the following command:

    ```
    AT+set=MQTT,flags,"url|sec|whitelist_rootca"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

20. To test if the connection was successful, we actually want to subscribe and publish to some MQTT topics. In order to be able to subscribe to a topic, we are setting the ‘subtopic0‘ parameter to "apple". This means that after connection, the Cordelia-I module will subscribe to the "apple" topic and listen for incoming messages on this topic. You can theoretically set all four subtopics from index 0 to 3, but in this tutorial, it is sufficient to use only one; the others can be left empty. To set ‘subtopic0‘ to "apple", go to the serial terminal program and send the following command:

    ```
    AT+set=SUBTOPIC0,name,"apple"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

21. In order to be able to publish to a topic, we also want to set the topic that we want to publish to. We want to publish to the "apple" topic, so that everything we publish is echoed back to the module. This way, we can verify that the module is working correctly. To achieve this, we can set the ‘pubtopic0‘ parameter to "apple". With this option, we can later publish messages to the "apple" topic. To set ‘pubtopic0‘ to "apple", go to the serial terminal program and send the following command:

    ```
    AT+set=PUBTOPIC0,name,"apple"
    ```

    If the setup was successful, the Cordelia-I module responds with an "OK" response.

22. Great! All the parameters required for a basic MQTT connection are set up. This means we are ready to initiate a connection. To do this, go to the serial terminal program and send the following command:

    ```
    AT+iotconnect
    ```

    If the connection was successful, the Cordelia-I module responds with an "OK" response, and after a few seconds, you should receive a "CONNACK" event, indicating that the connection to the MQTT broker was established successfully and you are ready to send and receive messages.

23. Now, let’s try to publish some messages! Go to the serial terminal program and send the following command:

    ```
    AT+iotpublish=0,"Hello␣Cordelia!"
    ```

    Remember, because we set the ‘pubtopic‘ with index 0 to "apple", and we are also subscribed to the "apple" topic, we should get the message back. And that is exactly what happens! If the publishing was successful, you first receive an "OK" response back from the Cordelia-I module, and after a short time, since we are subscribed to the "apple" topic, you should get a receive event containing the message you just sent out:

    ```
    +eventmqtt:recv,apple,qos0,"Hello␣Cordelia!"
    ```

24. Congratulations! You have successfully enrolled your Cordelia-I module to your QuarkLink instance and connected it using a secure MQTT connection. Feel free to play around, try other parameters, and send any kind of data you want.

14.5 Setting up your own basic non-secure MQTT broker and connecting a Cordelia-I module to it using manual AT commands

Setting up a non-secure MQTT broker on a Raspberry Pi provides a straightforward entry point into managing IoT devices and basic data communication without the complexity of encryption or client authentication. This setup is ideal for local network environments where security concerns are minimal, and it facilitates faster connectivity and simpler configurations. In this guide, we’ll install and configure Mosquitto, a lightweight and widely-used MQTT broker, on the Raspberry Pi, allowing the Cordelia-I module to connect to the broker via manual AT commands. With this approach, you’ll be able to send and receive MQTT messages on topics you define, enabling efficient device communication.

14.5.1 Setting up MQTT

Setting up a basic MQTT broker on your Raspberry Pi without TLS or authentication is straightforward. Here's a step-by-step guide using **Mosquitto**, a popular MQTT broker:

1. **Get IP Address**
   You should first retrieve and note the IP Address of your Raspberry Pi, because this address will be used throughout the process, in the common name, and during the connection.

   ```
   hostname -I
   ```

2. **Update Your Raspberry Pi**
   Open a terminal on your Raspberry Pi and run the following commands to ensure your system is up to date:

   ```
   sudo apt update
   sudo apt upgrade
   ```

3. **Install Mosquitto**
   Install Mosquitto and its clients with the following command:

   ```
   sudo apt install mosquitto mosquitto-clients
   ```

4. **Enable Mosquitto to Start on Boot**
   To ensure that the Mosquitto service starts automatically when your Raspberry Pi boots, run:

   ```
   sudo systemctl enable mosquitto
   ```

5. **Start Mosquitto**
   You can start the Mosquitto service using:

   ```
   sudo systemctl start mosquitto
   ```

6. **Verify the Installation**
   Check if Mosquitto is running with the following command:

   ```
   sudo systemctl status mosquitto
   ```

   You should see an output indicating that the service is active and running.

7. **Test the MQTT Broker**

   a) **Open a Terminal Window** (this can be done in a new terminal session).

   b) **Subscribe to a Topic:**

   ```
   mosquitto_sub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -t test/topic
   ```

   This command subscribes to the topic `test/topic`.

   c) **Publish a Message:**
   Open another terminal window and run:

   ```
   mosquitto_pub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -t test/topic -m "Hello␣MQTT"
   ```

   This command publishes the message `"Hello MQTT"` to the topic `test/topic`.

d) **Check the Output:**
The first terminal window (the subscriber) should display the message:

```
Hello MQTT
```

8. **Configure Mosquitto (Optional)**
By default, Mosquitto will work fine without any configuration for basic use. However, you can customize settings by editing the configuration file:

```
sudo nano /etc/mosquitto/mosquitto.conf
```

Here, you can change settings such as the default listener port or log locations. Save and exit the file (`Ctrl + X`, then `Y`, and `Enter`).

9. **Restart Mosquitto (if configured)**
If you made any changes to the configuration file, restart the Mosquitto service:

```
sudo systemctl restart mosquitto
```

That's it! You now have a basic MQTT broker running on your Raspberry Pi. You can connect your embedded devices or clients to this broker using the IP address of your Raspberry Pi.

14.5.2 Configuring the Cordelia-I module to connect to the broker

You should follow the same steps as in the example at Section 14.1, but for the iotHubEndpoint, you should set the IP Address of your Raspberry Pi. For the MQTT flags, you have to set "ip4" instead of "url".

14.6 Setting up your own secure MQTT broker and connecting a Cordelia-I module to it using manual AT commands

Implementing a secure MQTT broker using TLS on a Raspberry Pi allows for encrypted communication, protecting data integrity and confidentiality when devices connect over untrusted networks. This secure setup involves configuring Mosquitto to use a server certificate and a single root CA in PEM format, ensuring that data exchanged between the broker and client is securely encrypted. This guide covers each step to enable a TLS-based MQTT connection and configure the Cordelia-I module to authenticate the broker using the root CA without requiring a username and password. This setup balances security and manageability for IoT environments where device-to-cloud communication must be protected.

14.6.1 Setting up MQTT over TLS

To set up an MQTT broker on your Raspberry Pi over TLS with a single root CA in PEM format, open up a terminal window on your Raspberry Pi (or a similar Linux-based system) and follow these steps:

1. **Get IP Address**
You should first retrieve and note the IP Address of your Raspberry Pi, because this address will be used throughout the process, in the common name, and during the connection.

```
hostname -I
```

2. **Install Mosquitto Broker**
   Mosquitto is a lightweight MQTT broker that works well on a Raspberry Pi.

```
sudo apt update
sudo apt install mosquitto mosquitto-clients
```

3. **Create Certificates**
   You'll need a root CA and a server certificate signed by the root CA.

   a) **Generate the Root CA Certificate**

```
# Generate the root key
openssl genpkey -algorithm RSA -out ca.key
# Generate the root certificate
openssl req -new -x509 -days 3650 -key ca.key -out ca.crt -subj "/CN=MQTT␣Root␣CA"
```

   b) **Generate Server Certificate**

```
# Generate server key
openssl genpkey -algorithm RSA -out server.key
# Create a certificate signing request
openssl req -new -key server.key -out server.csr -subj "/CN=
    YOUR_RASPBERRY_PI_IP_ADDRESS"
# Sign the server certificate with the root CA
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out
    server.crt -days 365
```

   The files you'll have at the end are:

   - `ca.crt` (root CA certificate in PEM format to upload to your Cordelia-I embedded MQTT client)
   - `server.crt` and `server.key` (used by the broker)

4. **Configure Mosquitto for TLS**
   Edit the Mosquitto configuration file, usually found at `/etc/mosquitto/mosquitto.conf`:

```
# MQTT listener with TLS on port 8883
listener 8883
cafile /etc/mosquitto/ca_certificates/ca.crt
certfile /etc/mosquitto/certs/server.crt
keyfile /etc/mosquitto/certs/server.key

# Optional settings to enforce stronger security
tls_version tlsv1.2
require_certificate false

allow_anonymous true
```

5. **Add Certificates to Mosquitto Directory**
   Create the required directories and move the certificates there:

```
sudo mkdir -p /etc/mosquitto/certs /etc/mosquitto/ca_certificates
sudo cp server.crt server.key /etc/mosquitto/certs/
sudo cp ca.crt /etc/mosquitto/ca_certificates/
```

6. **Grant Permissions for Mosquitto**
   Make sure that `mosquitto` has permission to read the certificates:

```
sudo chown mosquitto:mosquitto /etc/mosquitto/certs/* /etc/mosquitto/ca_certificates/*
sudo chmod 640 /etc/mosquitto/certs/* /etc/mosquitto/ca_certificates/*
```

7. **Restart Mosquitto**

```
sudo systemctl restart mosquitto
```

8. **Test the MQTT Broker with TLS**
   Using `mosquitto_pub` and `mosquitto_sub`, test the setup locally to ensure that TLS is working:

```
# Subscribe to a test topic
mosquitto_sub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -p 8883 -t test/topic -d --cafile ca.crt
# Publish to the test topic
mosquitto_pub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -p 8883 -t test/topic -m "Hello over TLS"
    -d --cafile ca.crt
```

   You should now have an MQTT broker running on your Raspberry Pi over TLS, with the `ca.crt` as the root CA for your client.

Implications of `require_certificate false`

The line `require_certificate false` in the Mosquitto configuration means that the MQTT broker does *not* require a client certificate for clients to connect over TLS.

- **If `require_certificate` is set to `true`:** The broker will expect every client to provide a valid certificate signed by the same root CA as the server certificate. This enforces *mutual TLS (mTLS)*, where both the client and server authenticate each other. This is more secure but requires each client to have its own certificate.
- **If `require_certificate` is set to `false`:** The broker only requires the server certificate to establish a secure TLS connection, allowing clients to connect without a client certificate. Clients only need the *root CA certificate* to verify the broker's identity.

In most embedded applications where the broker verifies clients by username and password (or through some other means), `require_certificate false` is sufficient and easier to manage.

14.6.2 Adding authentication (username/password)

If you want to add username and password authentication in addition to TLS for your MQTT broker, you can configure Mosquitto to require authentication using a password file. Here's how to do it:

1. **Create a Password File**
   You can create a password file using the `mosquitto_passwd` utility that comes with the Mosquitto package.

   a) **Create the password file:**

```
sudo mosquitto_passwd -c /etc/mosquitto/passwd <username>
```

You will be prompted to enter a password for the specified username. Replace `<username>` with your desired username.

b) **Add additional users (optional):**
If you want to add more users later, use the command without the `-c` option:

```
sudo mosquitto_passwd /etc/mosquitto/passwd <another_username>
```

2. **Update the Mosquitto Configuration**
Edit your Mosquitto configuration file (usually `/etc/mosquitto/mosquitto.conf`) to include the following lines for authentication:

```
# Authentication settings
password_file /etc/mosquitto/passwd
allow_anonymous false
```

3. **Grant Permissions for Mosquitto**
Make sure that `mosquitto` has permission to read the certificates and password file:

```
sudo chown mosquitto:mosquitto /etc/mosquitto/certs/* /etc/mosquitto/ca_certificates/* /
    etc/mosquitto/passwd
sudo chmod 640 /etc/mosquitto/certs/* /etc/mosquitto/ca_certificates/* /etc/mosquitto/
    passwd
```

4. **Restart Mosquitto**
After updating the configuration file, restart the Mosquitto service to apply the changes:

```
sudo systemctl restart mosquitto
```

5. **Connecting with Username and Password**
When connecting to the MQTT broker with a client (such as using `mosquitto_pub` or `mosquitto_sub`), you can specify the username and password:

```
# Subscribe to a test topic with authentication
mosquitto_sub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -p 8883 -t test/topic -u <username> -P <
    password> --cafile ca.crt

# Publish to a test topic with authentication
mosquitto_pub -h <YOUR_RASPBERRY_PI_IP_ADDRESS> -p 8883 -t test/topic -m "Hello␣over␣TLS"
     -u <username> -P <password> --cafile ca.crt
```

Summary

Now, your MQTT broker will use TLS for secure communication and require clients to authenticate using a username and password. This provides an additional layer of security to your setup.

14.6.3 Configuring the Cordelia-I module to connect to the broker

You should follow the same steps as in the example at Section 14.2, but for the iotHubEndpoint, you should set the IP Address of your Raspberry Pi. For the MQTT flags, you have to set "ip4|sec|whitelist_rootca", and finally, instead of the test.mosquitto.org Root CA, you have to upload the Root CA of your own MQTT broker, then for the MQTT rootCAPath you have to set the path to the file that you just uploaded. Optionally, if you have set up username and password authentication, you also have to set the MQTT username and password user settings.

14.7 Connecting two Cordelia-I modules to each other in transparent mode

In this tutorial, we'll guide you through connecting two Cordelia-I modules in transparent mode to each other. The connection link can be established using any kind of MQTT connection, whether non-secure or secure. For simplicity, we'll use a basic non-secure connection on test.mosquitto.org. If you want to set up a secure connection, please refer to the other examples where the connection setup is explained in detail.

We'll use the serial terminal program HTerm to send commands and view responses from the module. Let's get started with setting up the module and establishing a wireless communication link!

1. Take two Cordelia-I evaluation boards and connect them to your PC with USB cables. By default, the devices start up in AT command mode, and they use virtual COM ports (115200 baud, 8N1) to communicate with the PC.

2. Follow one of the previous tutorials to set up an MQTT connection on both modules.

3. After ensuring that the MQTT connection on both modules is working, take the first module and set the pubtopic with index 0 and the subtopic with index 0. You can do this by sending the following commands:

   ```
   AT+set=PUBTOPIC0,name,"transparenttopicone"
   AT+set=SUBTOPIC0,name,"transparenttopictwo"
   ```

   If the setup was successful, the Cordelia-I module responds with an "OK" response for both commands.

4. Take the second module and set the pubtopic with index 0 and the subtopic with index 0. You can do this by sending the following commands:

   ```
   AT+set=PUBTOPIC0,name,"transparenttopictwo"
   AT+set=SUBTOPIC0,name,"transparenttopicone"
   ```

   Note that we changed the topic names to the opposite of the other module. This is because, with the first module, we want to receive payloads published by the second module, and with the second module, we want to receive payloads published by the first module. If the setup was successful, the Cordelia-I module responds with an "OK" response for both commands.

5. Now the two modules are set up in a way that in transparent mode, the first module is always publishing to the topic named "transparenttopicone", and the second module is always printing everything that is coming from the topic named "transparenttopictwo". Subsequently, the second module is always publishing to the topic named "transparenttopictwo", and the first module is always printing everything that is coming from the topic named "transparenttopicone". This means that there is a transparent connection between the two modules.

6. Take both modules and set the APP_MODE_0 and APP_MODE_1 pins to 1. On the Cordelia-I evaluation board, this corresponds to GPIO22 and GPIO0, which are Pin 12 and Pin 13 on the CON8 connector of the evaluation board. On the evaluation board,

this means taking a jumper wire and connecting those pins to VCC. This will make the modules boot up in transparent mode.

7. Press the Reset button on your evaluation boards. Now you don't get a startup message because the module boots up in transparent mode; instead, both LEDs on your evaluation boards should light up.

8. Open up two serial terminal programs, each for one Cordelia-I module. In this tutorial, we are using HTerm. Select the COM ports that your Cordelia-I evaluation boards are connected to, and open the COM ports to enable your computer to communicate with the modules. Set up the parameters of the COM ports to 115200 baud, 8 data bits, no parity bit, and 1 stop bit. In the transmit window, ensure that if you send a command, it is always terminated with a CR-LF sequence; otherwise, the Cordelia-I module might have problems parsing the payload.

9. For the first module, go to the serial terminal program and simply send the following sentence: "Hello Cordelia!". In a short time, the message should appear on the second module.

10. For the second module, go to the serial terminal program and do the same: simply send the following sentence: "Hello Cordelia!". In a short time, the message should appear on the first module.

11. Congratulations! You successfully connected two of your Cordelia-I modules to each other using a transparent MQTT connection. Feel free to play around, try other payloads, and send any kind of data you want.

15 Timing parameters

This section describes the behaviour of the Cordelia-I module during reset, sleep and wake-up operations.

15.1 Hard reset

A hard reset of the Cordelia-I module is done by asserting a low on the */RESET*. On hard reset, the module reloads the application from the sFlash after verifying the image to ensure the integrity of the application. This contributes towards higher start up times of the application.

Description	Typ.	Unit
Ready after reset	3500	ms

Table 66: Start-up time

15.2 Soft reset

A software reset is made available through the AT command `AT+reboot` (see section `11.1`). In this case the module restarts from the reset vector. The exact same process happens after a wake-up signal from sleep mode.

Description	Typ.	Unit
Ready after reboot/wake-up	2000	ms

Table 67: Start-up after reboot

It is recommended to use the AT command to reboot the device instead of a falling edge on the */Reset* pin whenever applicable.

Use `AT+stop` and `AT+start` to restart the network processor.

The fast/auto connect features ensure immediate connect to an AP on reboot/wake-up.

16 Firmware update

Cordelia-I supports secure firmware-over-the-air (FOTA) updates to enable easy update of the module's firmware in the field. The module needs to be connected to the internet via a local AP to enable firmware update in the field. In order to enable the module to perform FOTA updates, the following parameters need to be configured.

- **url**: is the URL of the FOTA server. This value by default is "https://www.we-online.com /res/wco/Cordelia".
- **rootCAPath**: is the path to the rootCA certificate (mandatory parameter for secure connections). The maximum length is 180 byte. The root CA certificate corresponding to the default FOTA server is present on the module by default.

By default, the module accesses the servers of Würth Elektronik eiSos to perform FOTA updates. However, in case the module does not have internet connectivity, the update packets can be hosted on local servers. In such cases, the above parameters needs to be configured accordingly.
The following steps are involved in performing a firmware update.

16.1 Check firmware version

In order to check the version of the firmware available, use the command described in section 11.2.1. This command creates a secure connection to the update server and fetches the version of the firmware update available. For example.

```
AT+get=ota, updateversion
+get:1.0.0
OK
```

Code 6: Example Get available version

Make sure that the module is connected to an AP which has access to the internet before executing this command.

Check the current version of the firmware installed on the module using the following command (see section 11.2.1).

```
AT+get=general, version
+get:0x31000019,2.7.0.0,2.2.0.7,3.20.0.1,0,0.9.0
OK
```

Code 7: Example Get installed version

The last parameter of the returned string is the current installed version of the firmware. In this example the version of the installed firmware is "0.9.0".

The firmware version check is optional and intended to inform the end user of the change in firmware. The module automatically checks the firmware version in the FOTA mode and performs an update only when a newer version of firmware is available. A downgrade to older version is not supported.

16.2 Update procedure

Restart the module in the OTA operating mode, by setting and holding *APP_MODE_0* and *APP_MODE_1* accordingly (see chapter 6.4.1). If correctly configured, the Cordelia-I automatically tries to connect to the AP previously configured. In case of the WLAN connection being successful, the *STATUS_IND_0* LED turns ON. In this case, the OTA procedure can be continued.

Further, the module connects to the configured update server and starts the update process. On successful start of the update process, the *STATUS_IND_0* LED turns ON.

Internally, the module performs a version check, downloads the update package, verifies the package for integrity and authenticity and finally updates the firmware on the module.
A successful firmware update process outputs the following logs on the UART. Details of the log may change with your module (e.g. MAC address and firmware version).

If the last event shows +eventota:error,"OTA_CANDIDATE_OLD",103 - than the firmware on the module is already up-to-date and the update was skipped. Which is intended behaviour in this case.

```
+eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
+eventwlan:connect,AP,0x34:0x31:0xc4:0x4a:0xeb:0x5f
+eventota:info,"CONNECTION_SUCCESS",0
+eventota:info,"OTA_START",0
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",04
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",08
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",12
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",16
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",20
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",24
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",28
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",32
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",36
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",40
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",44
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",48
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",52
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",56
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",60
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",64
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",68
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",72
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",76
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",80
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",84
```

```
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",88
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",92
+eventota:info,"DOWNLOAD_PROGRESS_PERCENT",96
+eventota:info,"DOWNLOAD_COMPLETED",1024
+eventota:info,"OTA_NOTIF_IMAGE_DOWNLOADED"
+eventota:info,"RESETTING_PLATFORM",0
```

Code 8: Example FOTA update

The module will reboot once after the FOTA is applied.

After this reset the module reboots. Since it will re-start in OTA mode (due to the pin setting) and it will detect that the FOTA update present in the memory of the module was already applied, it will inform the host with +eventota:error,"OTA_CANDIDATE_OLD",103 . This error is expected, since the module already has installed the firmware with this version.

```
+eventstartup:2610011025010,0x31000019,e4:15:f6:66:ca:45,0.7.2
+eventwlan:connect,WE_backup,0x34:0x31:0xc4:0x4a:0xeb:0x5f
+eventota:info,"CONNECTION_SUCCESS",0
+eventota:info,"OTA_START",0
+eventota:info,"NEW_IMAGE_COMMITTED",0
+eventota:error,"OTA_CANDIDATE_OLD",103
+eventota:error,"OTA_NOTIF_DOWNLOAD_ERROR",103
```

Code 9: Example reboot after a FOTA update

16.3 Update status indication

The *STATUS_IND_1* will be set to high on a successful WLAN connection. The *STATUS_IND_0* pin will be set to high once the FOTA update begins. The pins are set to low on reboot. The *STATUS_IND_1* will be set to high again on successful WLAN connection. The *STATUS_IND_0* pin will be set to high shortly indicating commit of the new image following which it will be set to low.

Do not interrupt the FOTA process when the *STATUS_IND_0* pin is high.

17 Firmware history

The Cordelia-I firmware is based on the SimpleLink WiFi CC3220 software development kit (SDK) from Texas Instruments with the corresponding features as well as known issues. A list of the versions of different components used for the current Cordelia-I firmware version is as shown below.

Description	Version
SimpleLink SDK Version (simplelink_cc32xx_sdk)	7.10.00.13
NWP Version	3.22.0.1
MAC Version	2.7.0.0
PHY Version	2.2.0.7
ROM Version	0

17.1 Release notes

Version 0.x.x "Engineering"

Version 1.0.0 "Release"

- First release of the product.

Version 1.1.0 "Service Pack Update"

- Updated NWP Version (Service Pack) from v3.20.0.1 to v3.22.0.1.

17.2 Known issues

Index	Details	Affected versions
KI-001	**Description:** When connecting over TLS1.2 (the "sec" option in the MQTT flags user setting is activated), and calling the command `AT+iotdisconnect`, the device might send an "OK" response twice. It is therefore recommended to configure your host controller to parse incoming serial data, and wait for the events signaling the end of the process (in this case, it is the MQTT event `CLIENT_DESTROY`).	1.0.0, 1.1.0

18 Hardware history

Version 2.2 "Release"

- Implementation of hardware history in the user manual.

19 Custom firmware and configuration

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

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

19.1 Custom configuration of standard firmware

The configuration of the standard firmware includes adoption of the non-volatile settings to customer requirements and creating a customized product based on the standard product.
This variant will result in a customer exclusive module with a unique ordering number. It will also freeze the firmware version to a specific and customer tested version and thus results in a customer exclusive module with a unique ordering number.
Further scheduled firmware updates of the standard firmware will not be applied to this variant automatically. Applying updates or further functions require a customer request and release procedure.

19.2 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 I^2C sensor) and cyclic transmission of this data to a data collector, while sleeping or being passive most of the time.
Also replacing UART with SPI as host communication interface is classified as a custom specific option.
Certification critical changes need to be re-evaluated by an external qualified measurement laboratory. These critical changes may occur when e.g. changing radio parameters, the channel access method, the duty-cycle or in case of various other functions and options possibly used or changed by a customer specific firmware.

19.3 Customer firmware

A customer firmware is a firmware written and tested by the customer himself or a 3rd party as a customer representative specifically for the hardware platform provided by a module.
This customer firmware (e.g. in form of an Intel hex file) will be implemented into the module's production process at our production site.

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

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

20 Design in guide

20.1 Advice for schematic and layout

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

The following general advice should be taken into consideration:

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

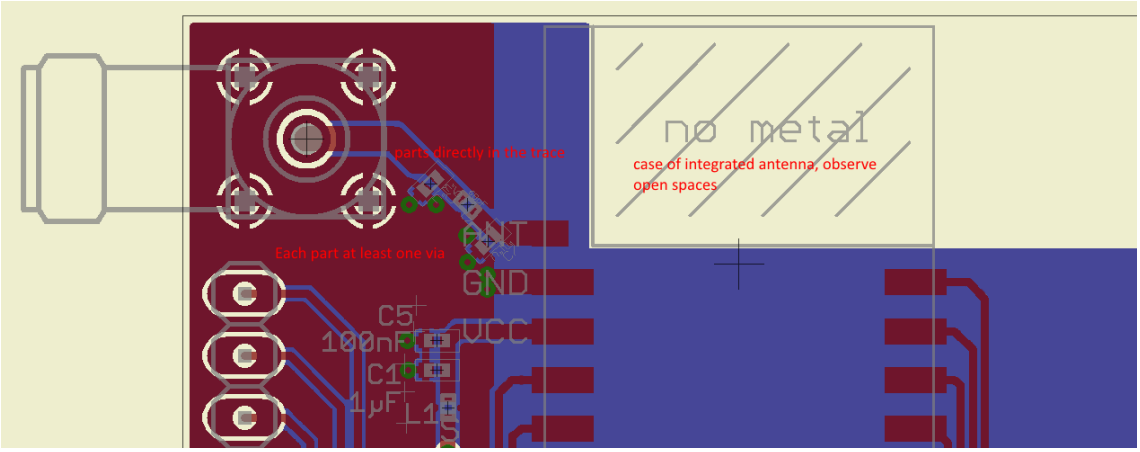


Figure 31: Layout

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

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

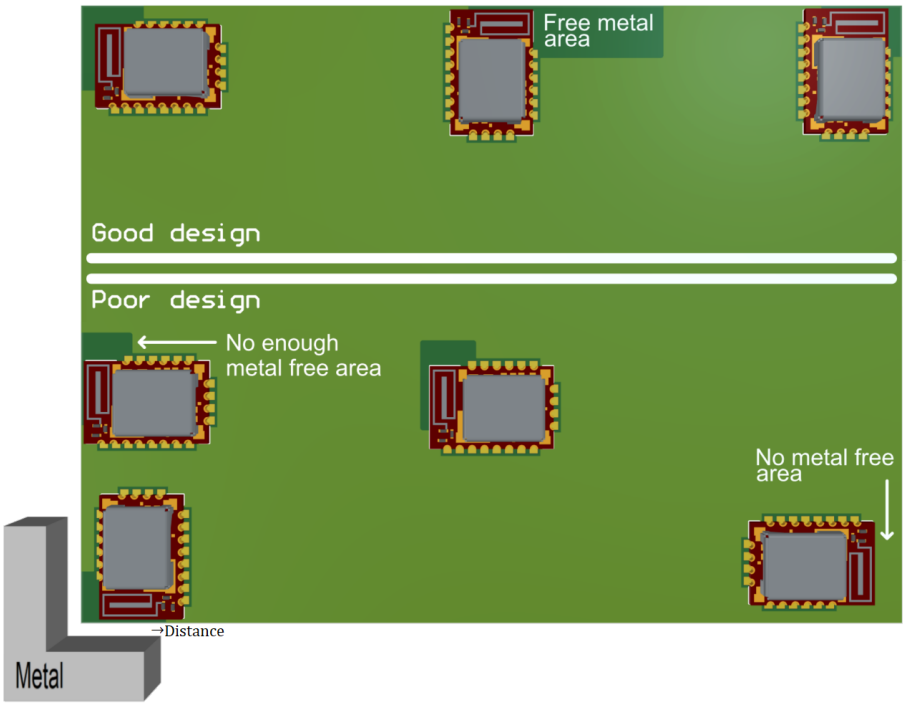


Figure 32: Placement of the module with integrated antenna

20.2 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.

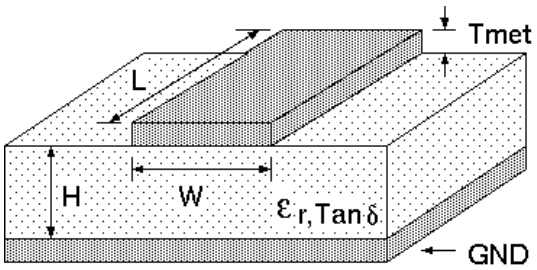


Figure 33: Dimensioning the antenna connection as micro strip

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

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

Example:
A FR4 material with ε_r = 4.3, a height H = 1000 µm and a copper thickness of T_{met} = 18 µm will lead to a trace width of W ~ 1.9 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×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.

20.3 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 λ /10 (which is 3.5 cm @ 868 MHz and 1.2 cm @ 2.44 GHz) from the antenna to any other metal should be kept. Metal placed further away will not directly influence the behavior of the antenna, but will anyway produce shadowing.

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

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

20.3.1 Wire antenna

An effective antenna is a λ /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 λ/4 radiator has approximately 40 Ω input impedance. Therefore, matching is not required.

20.3.2 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.

20.3.3 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.

20.3.4 Antennas provided by Würth Elektronik eiSos

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

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

20.3.4.1 2600130021 - Himalia dipole antenna

Figure 34: 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. [°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.

21 Reference design

Cordelia-I was tested and certified on the corresponding Cordelia-I EV-Board. For the compliance with the EU directive 2014/53/EU Annex I, the EV-Board serves as reference design. For the FCC it serves as trace design.
This is no discrepancy due to the fact that the EV-Board itself does not fall within the scope of the EU directive 2014/53/EU Annex I as the module is tested on the EV-Board, which is also the recommended use.

Further information concerning the use of the EV-Board can be found in the manual of the Cordelia-I EV-Board.

Calypso and Cordelia-I are hardware identical in terms of radio module and evaluation board.

21.1 EV-Board

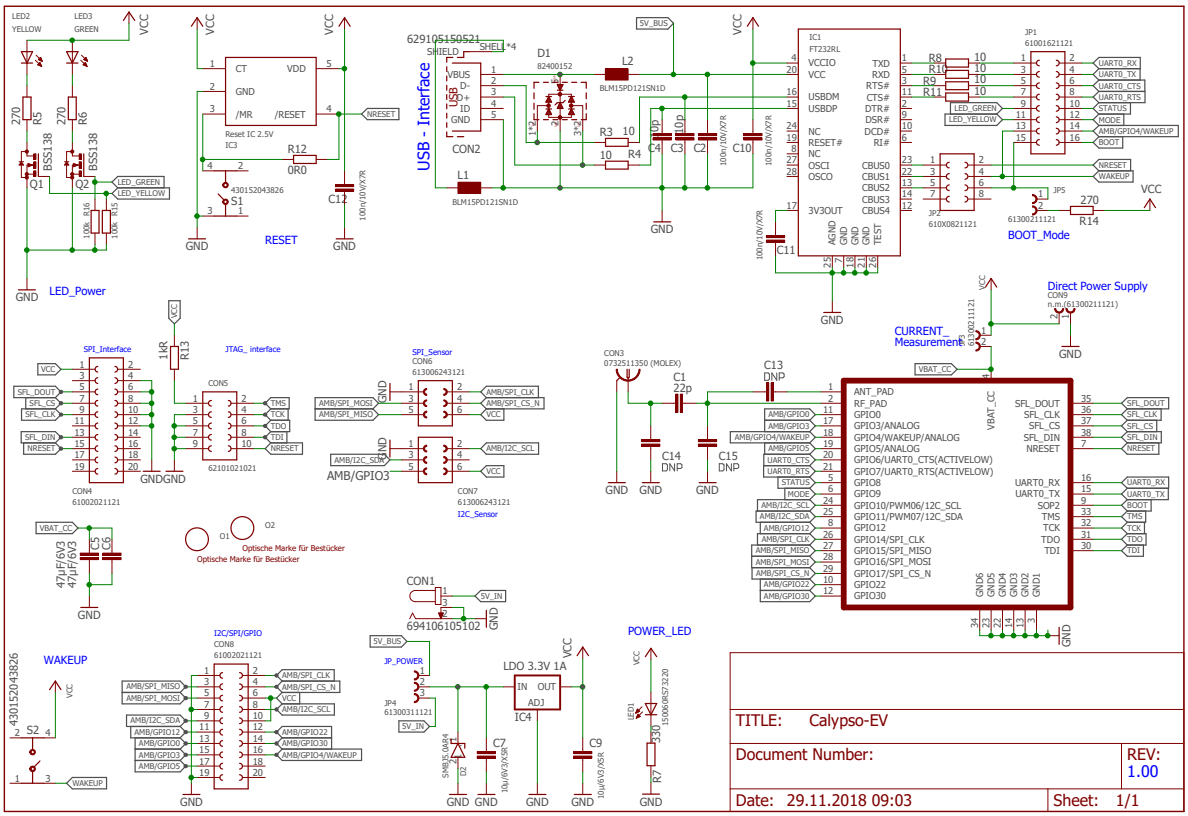


Figure 35: Reference design: Schematic, most important parts

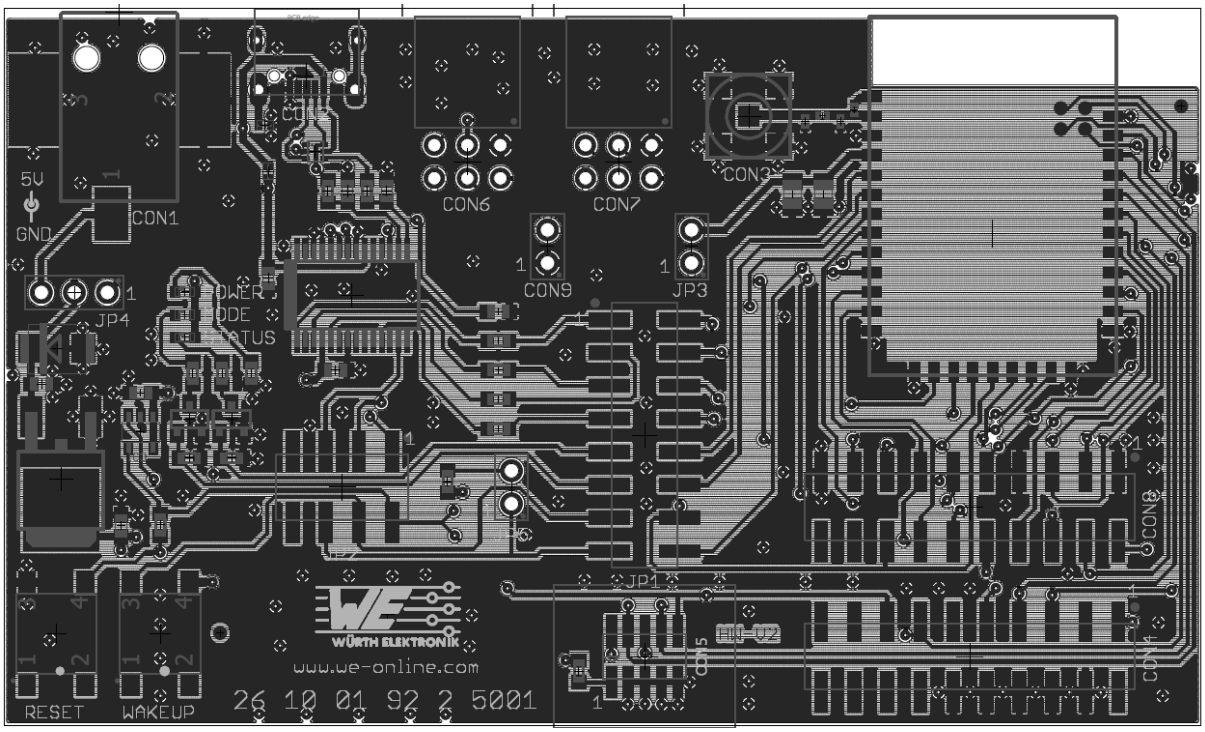


Figure 36: Reference design: Layout

21.2 Radiation characteristic of the module's internal antenna

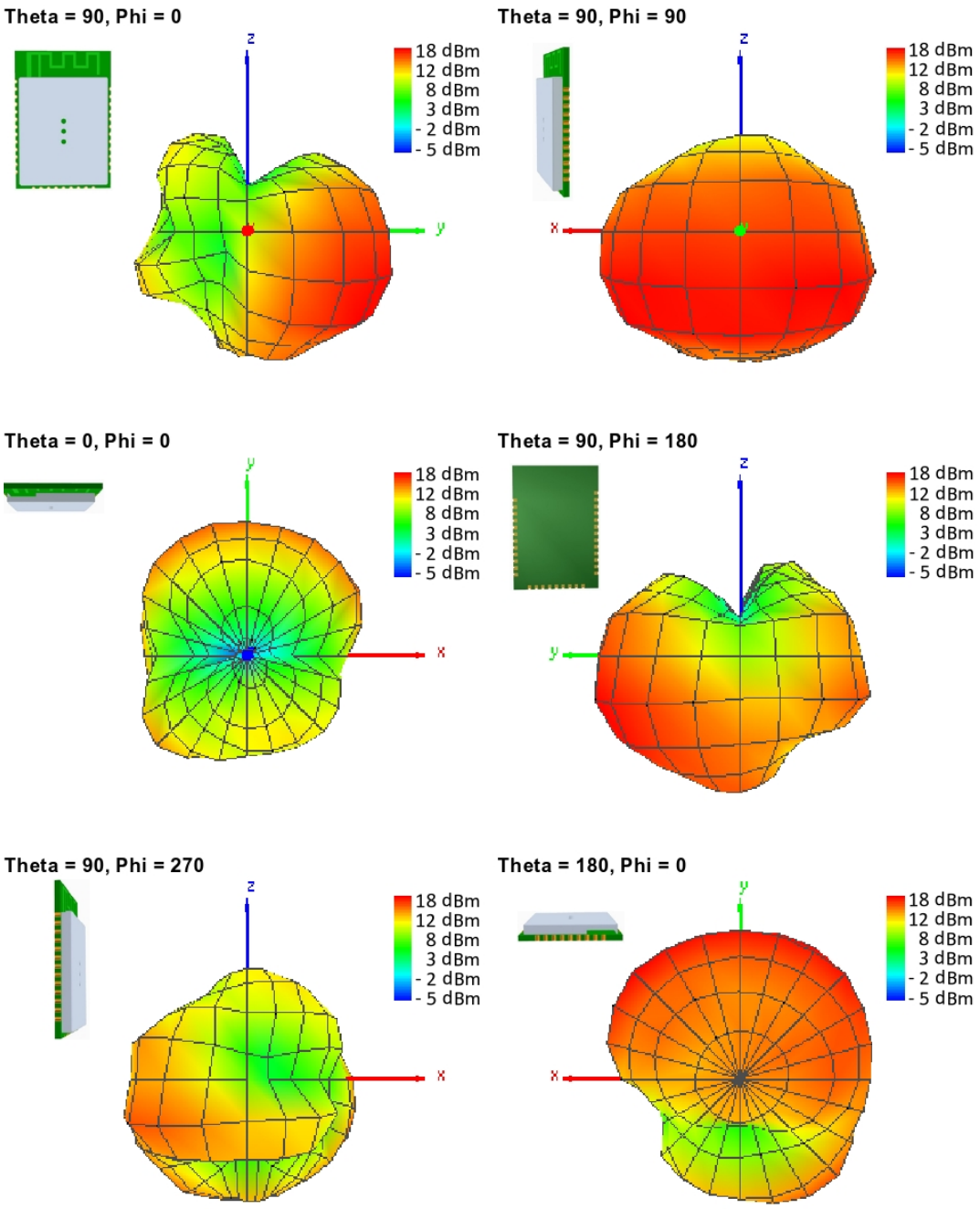


Figure 37: Antenna characteristic of the module with its integrated antenna measured on the official EV-Board

It is important to be aware that size and shape of the ground plane as well as the placement of module has influence on the radiation pattern.

21.3 Design Guide for FCC ID R7T1001102

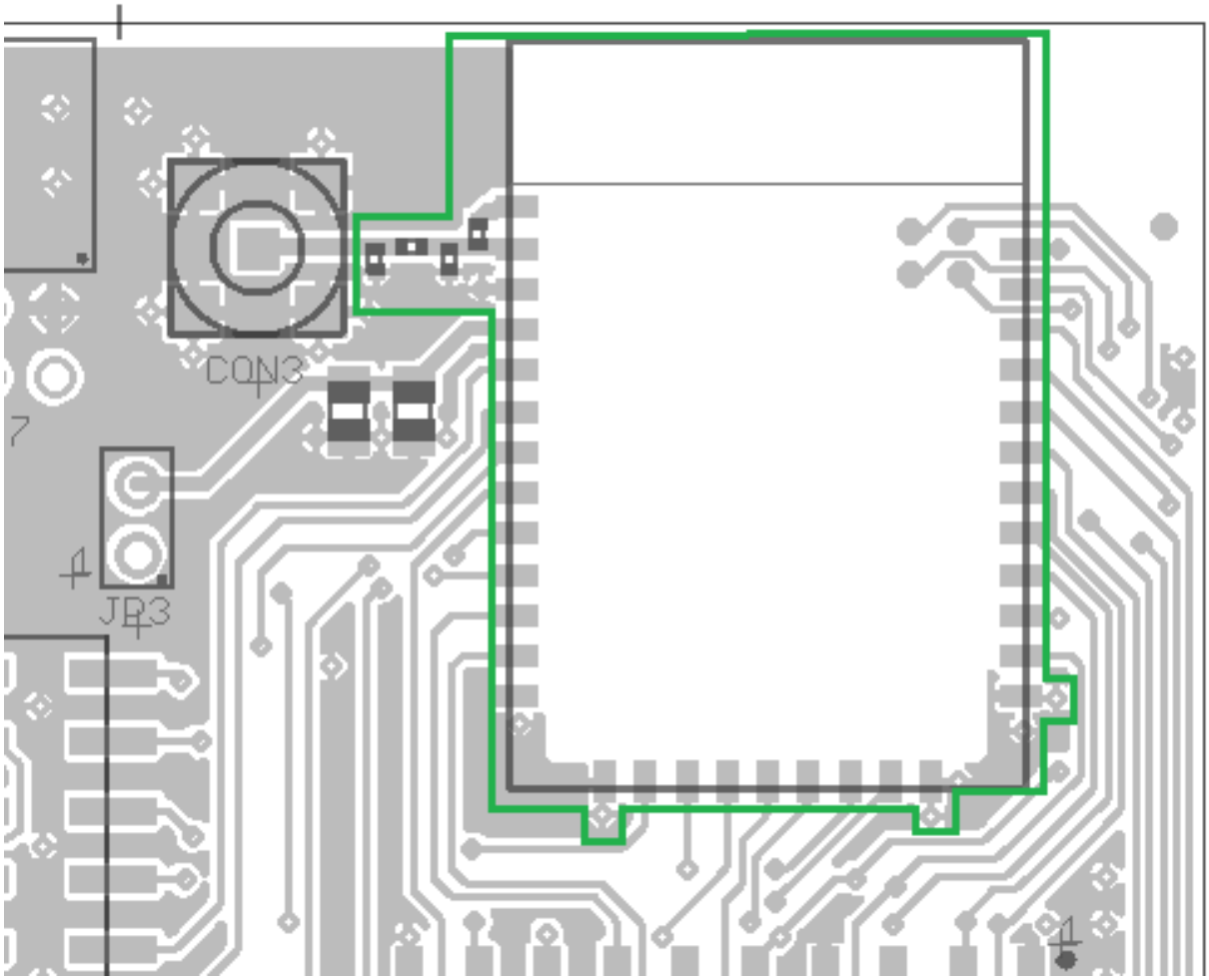


Figure 38: Close-up: Layout

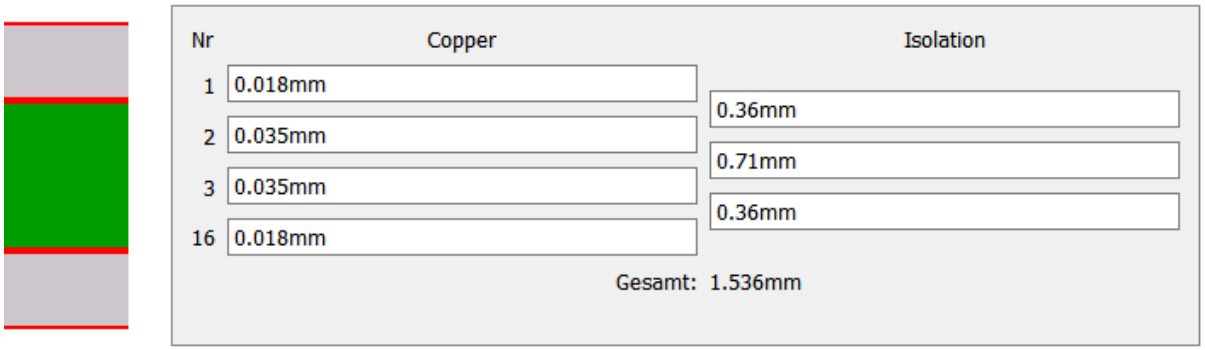


Figure 39: Reference design: Stack-up

- Top layer is used for routing and filled up with ground except underneath the module and the antenna free area.

- Second layer is ground, except the antenna free area.
- Third layer is the supply layer, except antenna free area. Some routing is allowed, not dividing the supply layer in to many or to small parts.
- Bottom layer is used for routing.

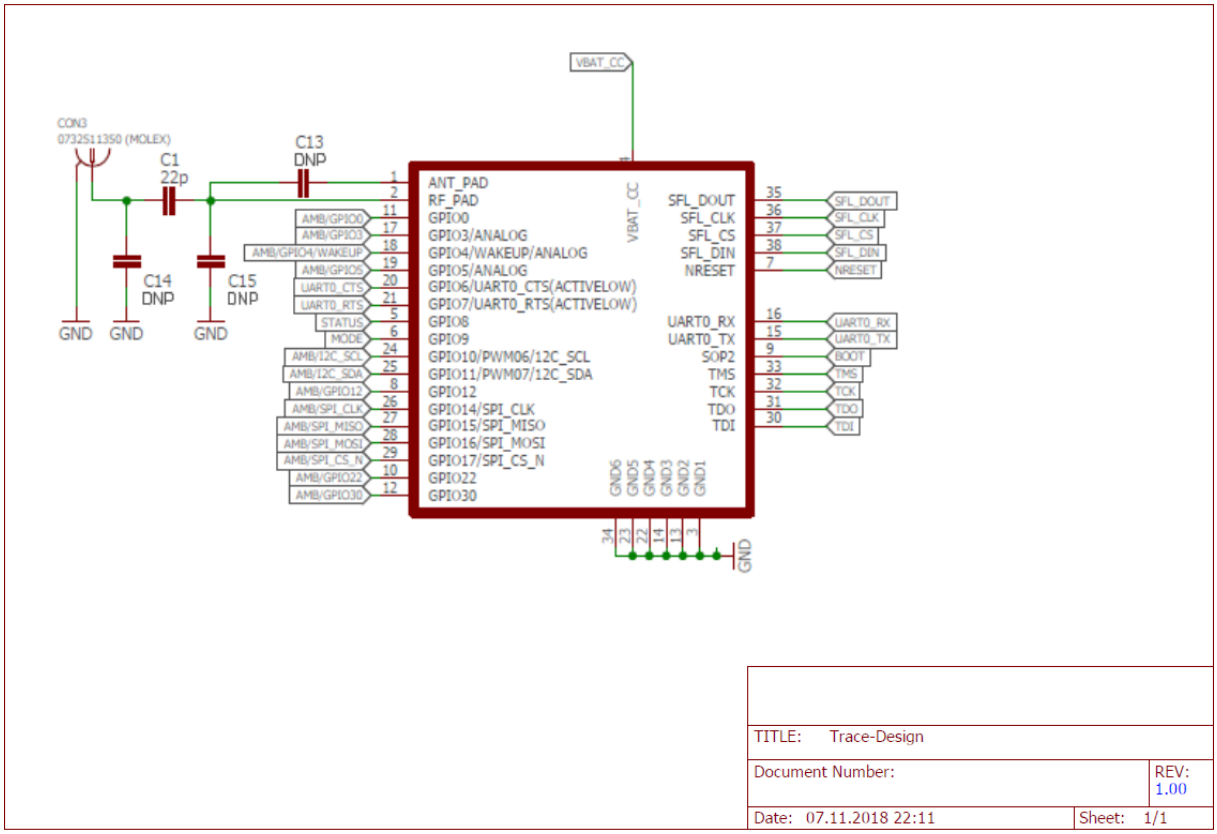


Figure 40: Close-up: Schematic

Two variants of the Cordelia-I are certified:

- **Integrated PCB antenna:** Not placing C1, C14 and C15, but placing 0 Ω at C13. C13 connects the *RF* pad, the radio signal to/from the transceiver, to the *ANT* pad, the connection to the module's integrated PCB antenna.
- **External antenna:** Placing 22pF at C1, not placing C13, C14 and C15 and connecting C1 with a 50 Ω line to a dipole antenna. For the certification the antenna from `20.3.4.1` was used with a peak gain of 2.8 dBi.

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

21.4 Application mode pins

The pins *APP_MODE_0* and *APP_MODE_1* define at boot time which application mode is used during operation of the module (see chapter 6.4.1).
The OTA mode enables security updates of the firmware and/or HTTP server certificates via radio and provisioning mode may be used for configuring the module.

To manually switch to OTA mode or provisioning mode, it is strongly recommended to make the pins *APP_MODE_0* and *APP_MODE_1* accessible on the custom PCB. Otherwise the update of certificates and the firmware is not possible.

22 Manufacturing information

22.1 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*.

22.2 Soldering

22.2.1 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	$T_{S\ Min}$	150 °C
Preheat temperature, max	$T_{S\ Max}$	200 °C
Preheat time from $T_{S\ Min}$ to $T_{S\ Max}$	t_S	60 - 120 s
Ramp-up rate (T_L to T_P)		3 °C/s max.
Liquidous temperature	T_L	217 °C
Time t_L maintained above T_L	t_L	60 - 150 s
Peak package body temperature	T_P	245 °C
Time within 5 °C of actual peak temperature	t_P	20 - 30 s
Ramp-down rate (T_P to T_L)		6 °C/s max.
Time 20 °C to T_P		8 min max.

Table 69: 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.

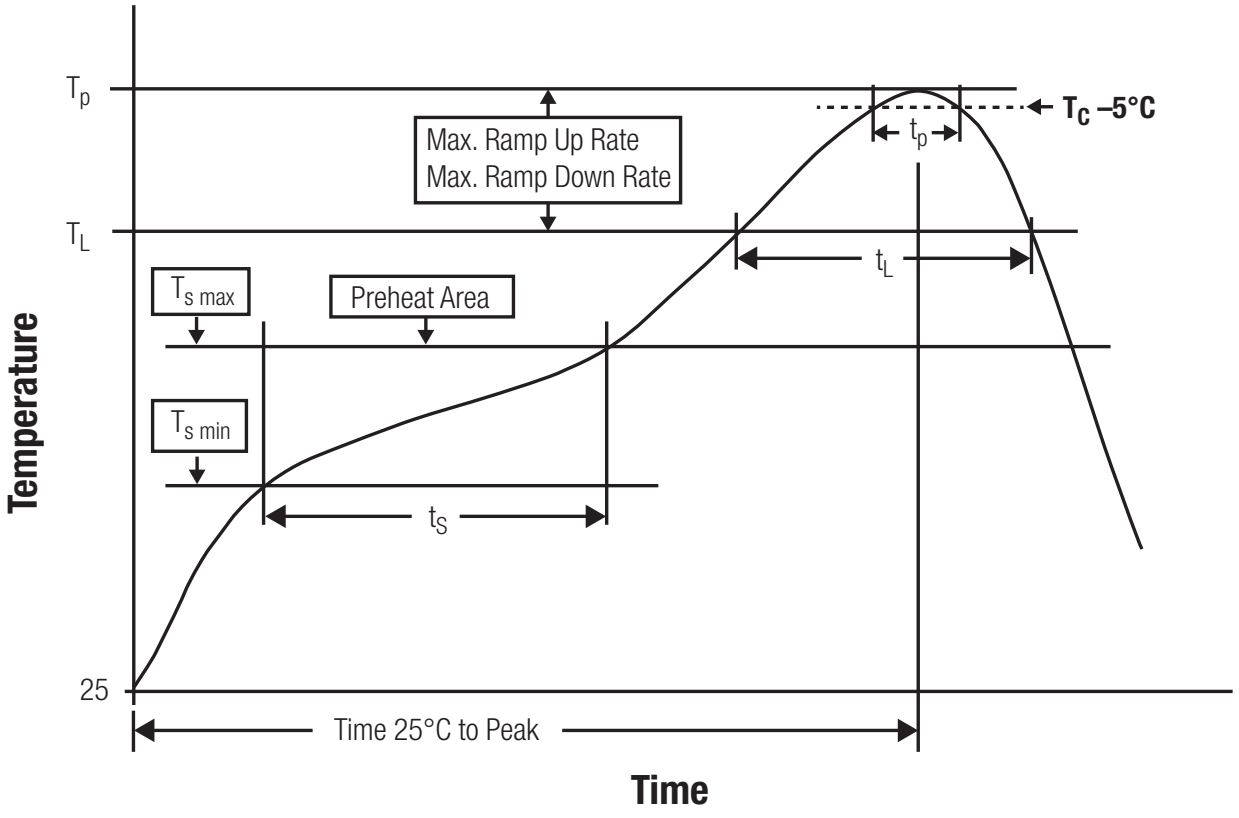


Figure 41: Reflow soldering profile

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

22.2.2 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.

22.2.3 Potting and coating

- If the product is potted in the customer application, the potting material might shrink or expand during and after hardening. Shrinking could lead to an incomplete seal, allowing contaminants into the component. Expansion could damage components. We recommend a manual inspection after potting to avoid these effects.
- Conformal coating or potting results in loss of warranty.
- The RF shield will not protect the part from low-viscosity coatings and potting. An undefined amount of coating and potting will enter inside the shielding.
- Conformal coating and potting will influence the parts of the radio front end and consequently influence the radio performance.
- Potting will influence the temperature behavior of the device. This might be critical for components with high power.

22.2.4 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.

22.3 ESD handling

This product is highly sensitive to electrostatic discharge (ESD). As such, always use proper ESD precautions when handling. Make sure to handle the part properly throughout all stages of production, including on the host PCB where the module is installed. For ESD ratings, refer to the module series' maximum ESD section. For more information, refer to the relevant chapter 4. Failing to follow the aforementioned recommendations can result in severe damage to the part.

- The first contact point when handling the PCB is always between the local GND and the host PCB GND, unless there is a galvanic coupling between the local GND (for example work table) and the host PCB GND.
- Before assembling an antenna patch, connect the grounds.
- While handling the RF pin, avoid contact with any charged capacitors and be careful when contacting any materials that can develop charges (for example coaxial cable with around 50-80 pF/m, patch antenna with around 10 pF, soldering iron etc.)
- Do not touch any exposed area of the antenna to avoid electrostatic discharge. Do not let the antenna area be touched in a non ESD-safe manner.
- When soldering, use an ESD-safe soldering iron.

22.4 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.

23 Product testing

23.1 Würth Elektronik eiSos in-house production tests

To achieve a high quality standard, Würth Elektronik eiSos follows a philosophy of supplying fully tested radio modules. At the end of the production process, every unit undergoes an optical inspection. Here the quality of soldering, edge castellation and edge milling is monitored.

If this has been passed, the radio modules are handed over to the automatic test equipment for the electrical characterization. This includes:

- Voltage and current tests to ensure proper electrical performance
- RF characteristics (frequency, spectrum, TX power) measurement and calibration
- Radio communication tests
- Firmware and serial number programming
- Host interface communication tests

The automated testing process is logged for internal quality control. The gained measurement data of each unit is analysed to detect defective parts and investigate the corresponding root cause. Defective radio modules are discarded, in order to guarantee a 100% failure-free delivery to customers.

23.2 EMS production tests

The rigorous in-series production testing ensures that EMS don't need to duplicate firmware tests or measurements. This streamlines the process and eliminates the need for additional testing over analogue and digital interfaces during device production. When it comes to device testing, the ideal focus should be on module assembly quality:

- All module pins are soldered properly on the base PCB
- There are no short circuits
- The mounting process did not damage the module
- The communication between host and radio module is working
- The antenna is connected properly

Simple "Go/No go" tests, like checking the RSSI value, give already a hint if the power supply and antenna have been connected properly.

In addition to such standard testing procedures, radio module integrators have the flexibility to perform additional dedicated tests to thoroughly evaluate the device. Specific tests they can consider are:

- Measure module current consumption in a specified operating state. Deviations from expected results (compared to a "Golden Device") can signal potential issues.

- Perform functional tests, including communication checks with the host controller and verification of interfaces.
- Assess fundamental RF characteristics (modulation accuracy, power levels, spectrum). Verify that the device meets expected performance standards.

24 Physical specifications

24.1 Dimensions

Dimensions
19 * 27.5 * 3 mm

Table 70: Dimensions

24.2 Weight

Weight
3 g

Table 71: Weight

24.3 Module drawing

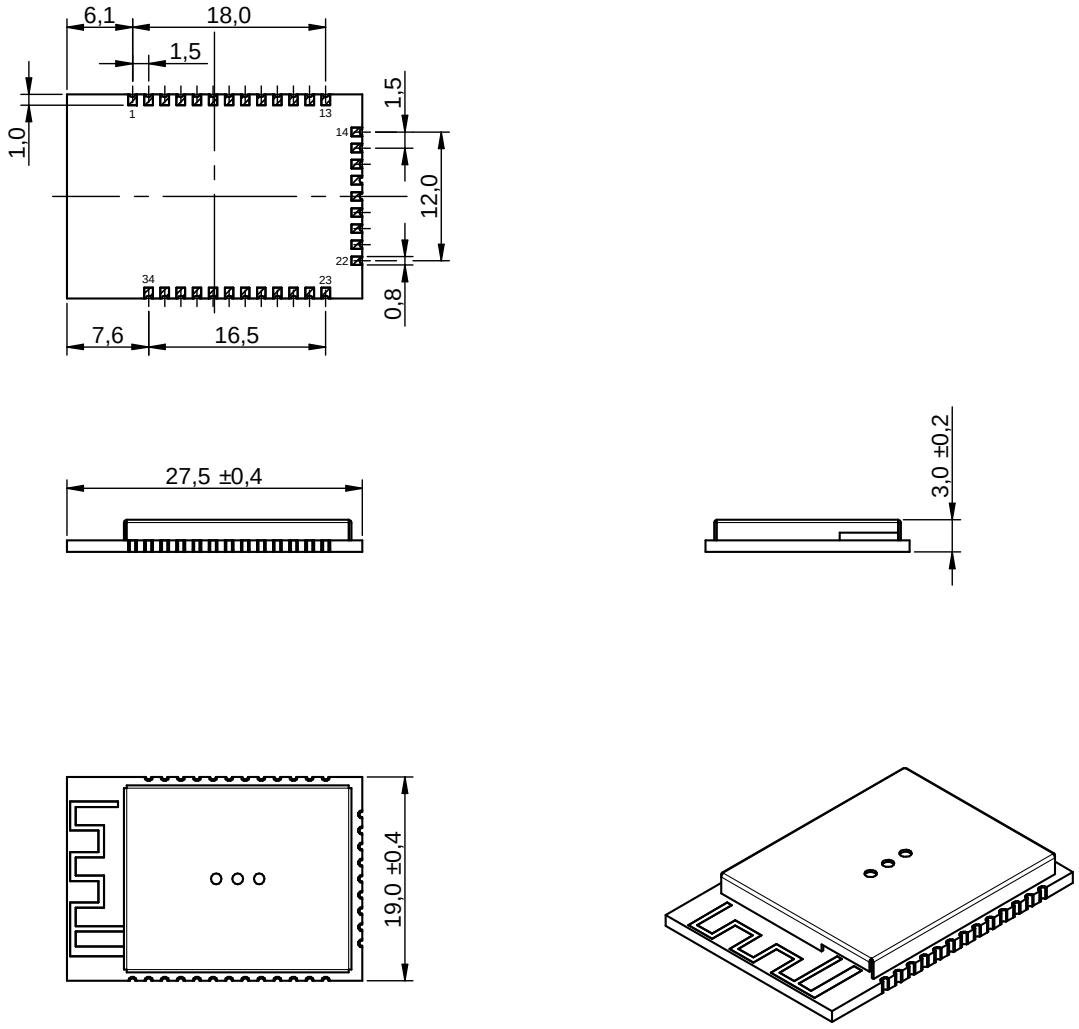


Figure 42: Module dimensions [mm]

24.4 Footprint WE-FP-5

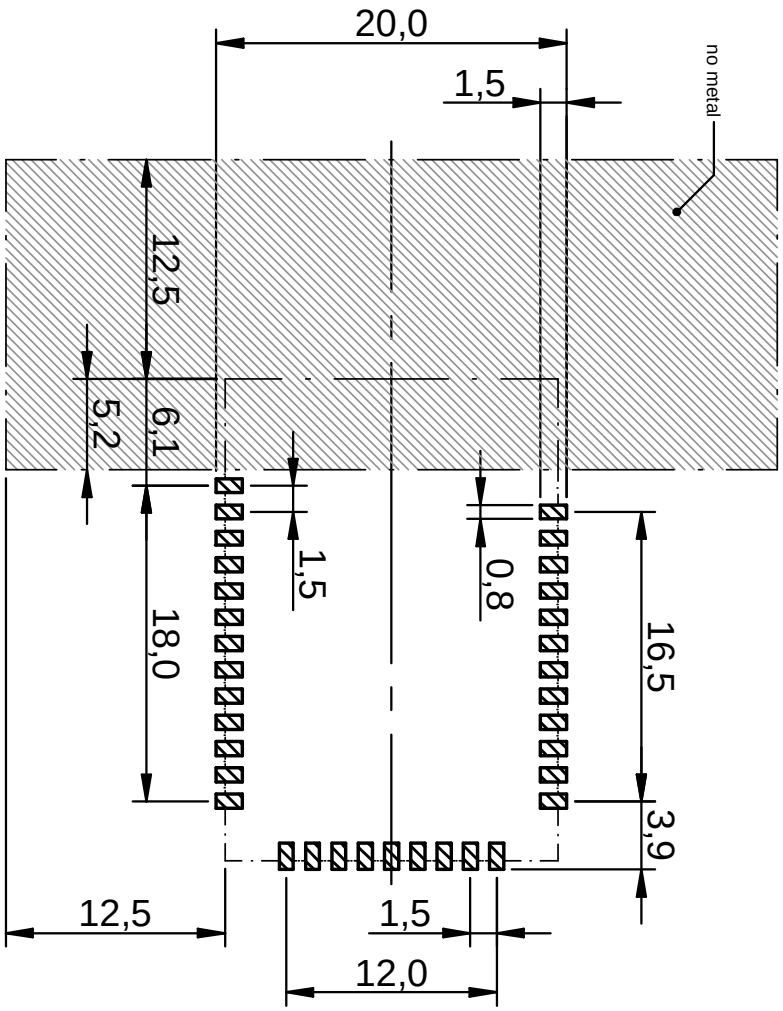


Figure 43: Footprint WE-FP-5 and dimensions [mm]

24.5 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 λ/10 should be kept (see figure 43). Even though metal parts would influence the characteristic of the antenna, but the direct influence and matching keep an acceptable level.

25 Marking

25.1 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.

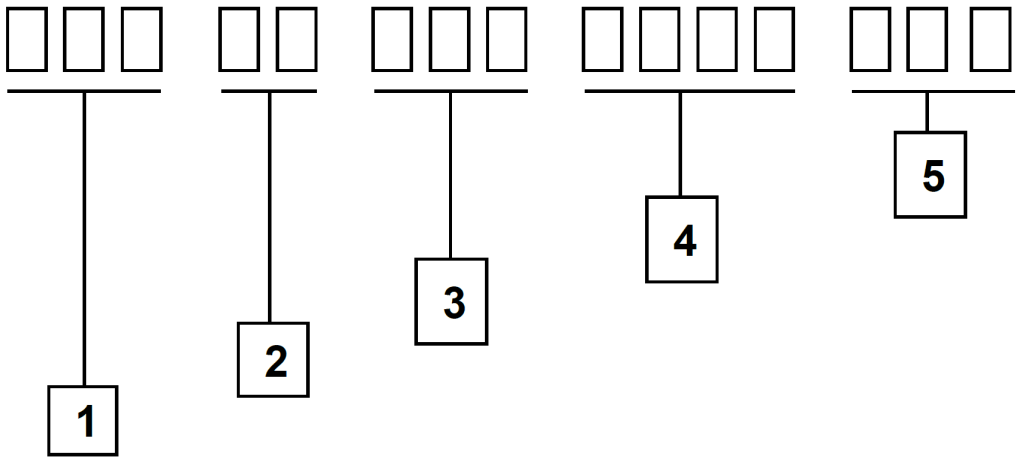


Figure 44: Lot number structure

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 72: Lot number details

As the user can perform a firmware update the printed lot number only shows the factory delivery state. The currently installed firmware can be requested from the module using the corresponding product specific command. The firmware version as well as the hardware version are restricted to show only major and minor version not the patch identifier. Block 5 is not applicable for products without firmware.

25.2 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 Cordelia-I the label is as follows:

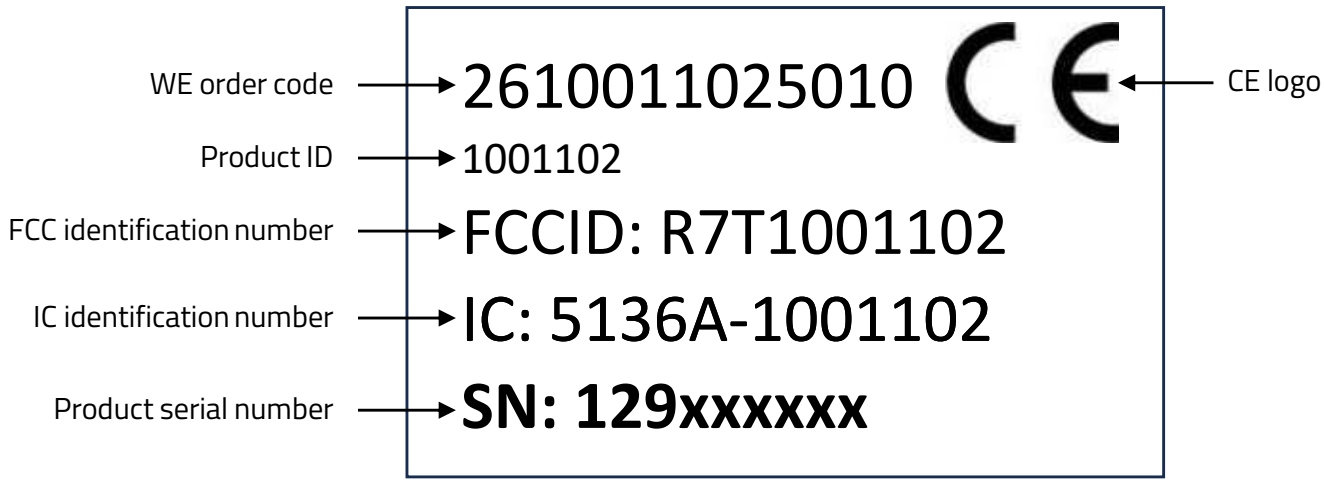


Figure 45: Label of the Cordelia-I

26 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 has no internal fuse.
- The maximum output power of the module is 18 dBm.
- The total amount of capacitance of all capacitors is 91.1 µF.
- The total amount of inductance of all inductors is 15.4 µH.

27 Regulatory compliance information

27.1 Important notice EU

The use of RF frequencies is limited by national regulations. The Cordelia-I has been designed to comply with the RED directive 2014/53/EU of the European Union (EU).
The Cordelia-I can be operated without notification and free of charge in the area of the European Union. However, according to the RED directive, restrictions (e.g. in terms of duty cycle or maximum allowed RF power) may apply.

Modifications (2014/53/EU article 3 (i))
Caution: Changes or modifications for this equipment not expressly approved by Würth Elektronik eiSos may void the CE conformity to operate this equipment.

27.2 EU Declaration of conformity

EU DECLARATION OF CONFORMITY

Radio equipment: **Cordelia-I / 2610011025010**

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: Cordelia-I / 2610011025010

The object of the declaration described above is in conformity with the relevant Union harmonisation legislation Directive 2014/53/EU. Following harmonised norms or technical specifications have been applied:

EN 300 328 V2.2.2 (2019-07)
EN 18031-1:2024
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 + A11:2017
2011/65/EU with its amending Annex II EU 2015/863
EN IEC 63000:2018

i.A. G. Eckhardt

Gudrun Eckhardt, Teamleader Hardware Development WCS, Trier, 7th of July 2025
Name, function, place and date of issue

27.3 RED-DA Cybersecurity statement

Cybersecurity as per articles 3.3d, 3.3e and 3.3f of the Radio Equipment Directive Delegated Act. Compliance to RED-DA is achieved by complying to the EN 18031-1, 18031-2 and 18031-3.

- EN 18031-1: Common security requirements for radio equipment - Part 1: Internet connected radio equipment
- EN 18031-2: Common security requirements for radio equipment - Part 2: radio equipment processing data, namely Internet connected radio equipment, childcare radio equipment, toys radio equipment and wearable radio equipment
- EN 18031-3: Common security requirements for radio equipment - Part 3: Internet connected radio equipment processing virtual money or monetary value

The Cordelia-I cannot interact with the internet in factory delivery state. The end-device is required to configure WLAN credentials (SSID, password, security method) to allow a WLAN connection, thus enabling the connection to the internet.

Requirements	Statement and conditions
(d) Radio equipment does not harm the network or its functioning nor misuses network resources, thereby causing an unacceptable degradation of service	"Applicable": The product implements the following internet-connectable protocols. IEEE 802.11bgn Client (WLAN) MQTT over TLS Client for cloud connectivity HTTPS Client for FOTA HTTPS Client to QuarkLink™ SNTP Client mDNS Client DHCP Client ARP ICMP The product implements the following host interfaces. UART, AT command based
(e) Radio equipment incorporates safeguards to ensure that the personal data and privacy of the user and of the subscriber are protected	"Not applicable": The product does not pose a risk to the users or subscribers privacy, as it does not store or process any personal data.

(f) Radio equipment supports certain features ensuring protection from fraud	"Not applicable": The product does not pose a risk of fraud because it does not store or process financial data or enables financial transactions.

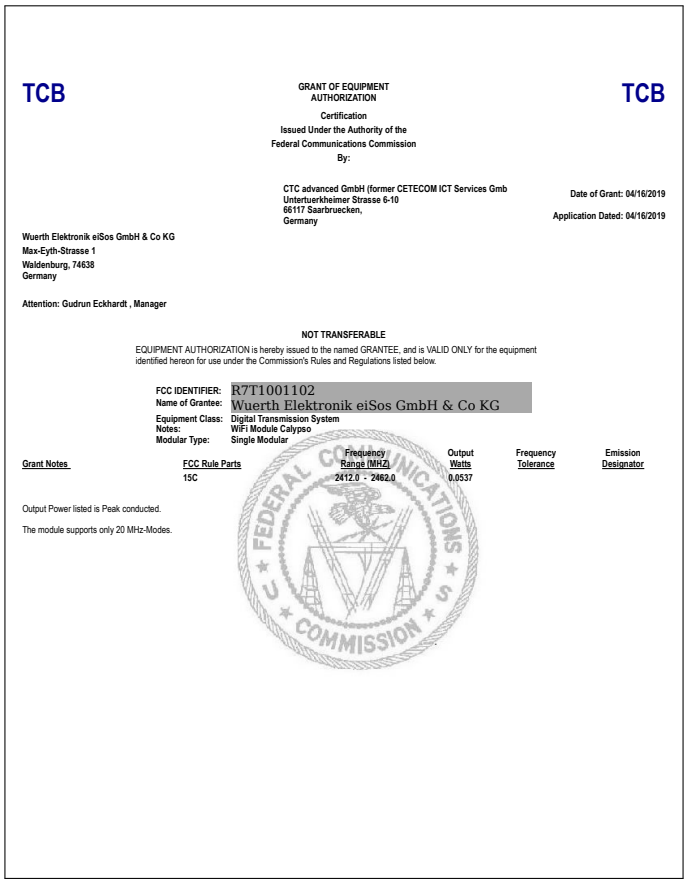
27.4 FCC Compliance Statement (US)

FCC ID: R7T1001102

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.

27.4.1 FCC certificate



TCB

GRANT OF EQUIPMENT AUTHORIZATION

Certification
Issued Under the Authority of the
Federal Communications Commission
By:

TCB

CTC advanced GmbH (former CETECOM ICT Services Gmb
Untertuerkheimer Strasse 6-10
66117 Saarbruecken,
Germany

Date of Grant: 04/16/2019
Application Dated: 04/16/2019

Wuerth Elektronik eiSos GmbH & Co KG
Max-Eyth-Strasse 1
Waldenburg, 74638
Germany

Attention: Gudrun Eckhardt , Manager

NOT TRANSFERABLE

EQUIPMENT AUTHORIZATION is hereby issued to the named GRANTEE, and is VALID ONLY for the equipment identified hereon for use under the Commission's Rules and Regulations listed below.

FCC IDENTIFIER: R7T1001102
Name of Grantee: Wuerth Elektronik eiSos GmbH & Co KG
Equipment Class: Digital Transmission System
Notes: WiFi Module Calypso
Modular Type: Single Modular

Grant Notes	FCC Rule Parts	Frequency Range (MHZ)	Output Watts	Frequency Tolerance	Emission Designator
	15C	2412.0 - 2462.0	0.0537		

Output Power listed is Peak conducted.

The module supports only 20 MHz-Modes.

FEDERAL COMMUNICATIONS COMMISSION

Figure 46: FCC certificate

27.5 IC Compliance Statement (Canada)

Certification Number: 5136A-1001102
HVIN: 1001102
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.

27.5.1 IC certificate


Technical Acceptance Certificate - Canada

CTC advanced
member of RWTÜV group

Certificate Holder: Würth Elektronik eiSos GmbH & Co
Max-Eyth-Str. 1
74638 Waldenburg
Germany

Bundesnetzagentur
BNetzA-CAB-03/22-51
authorized by the German Government to act as CAB in accordance with the MRA EU Canada of 1st November 1998.

ISED Certification Number: 5136A-1001102

CTC Registration Number: 2113

OATS Facility ID Number: 3462C

OATS Facility: CTC advanced GmbH
Untertuerkheimer Str. 6 -10
66117 Saarbruecken
Germany
Phone: +49 681 598-0
Fax: +49 681 598-8775
Email: info@ctcadvanced.com

Product Description: WiFi Module Calypso

Certification of equipment means only that the equipment has met the requirements of the above-noted specification. Licence applications, where applicable to use certified equipment, are acted on accordingly by the ISED issuing office and will depend on the existing radio environment, service and location of operation. This certificate is issued on condition that the holder complies and will continue to comply with the requirements and procedures issued by ISED. The equipment for which this certificate is issued shall not be manufactured, imported, distributed, leased, offered for sale or sold unless the equipment complies with the applicable technical specifications and procedures issued by ISED.

La certification du matériel signifie seulement que le matériel a satisfait aux exigences de la norme indiquée ci dessus. Les demandes de licences nécessaires pour l'utilisation du matériel certifié sont traitées en conséquence par le bureau de délivrance d'ISDE et dépendent des conditions radio ambiantes, du service et de l'emplacement d'exploitation. Le présent certificat est délivré à la condition que le titulaire satisfasse et continue de satisfaire aux exigences et aux procédures d'ISDE. Le matériel à l'égard duquel le présent certificat est délivré ne doit pas être fabriqué, importé, distribué, loué, mis en vente ou vendu à moins d'être conforme aux procédures et aux spécifications techniques applicables publiées par ISDE.

I hereby attest that the subject equipment was tested and found in compliance with the above-noted specification.
J'atteste par la présente que le matériel a fait l'objet d'essai et jugé conforme à la spécification ci-dessus.

FOREIGN CERTIFICATION BODY
CAB ID NO DE0001

CTC advanced GmbH
cn=Stefan Boes, o=CTC advanced GmbH, ou=BOE-161129, email=Stefan.Boes@ctcadvanced.com, c=DE
2019.04.16 15:44:44 +02'00'

Saarbrücken
Certification Body

CTC advanced GmbH (formerly CETECOM ICT Services GmbH)
Untertuerkheimer Str. 6–10 | 66117 Saarbrücken | Germany | www.ctcadvanced.com

Figure 47: IC certificate

27.6 FCC and IC requirements to OEM integrators

This module has been granted modular approval. OEM integrators for host products may use the module in their final products without additional FCC/IC (Industry Canada) certification if they meet the following conditions. Otherwise, additional FCC/IC approvals must be obtained. The host product with the module installed must be evaluated for simultaneous transmission requirements.

- The users manual for the host product must clearly indicate the operating requirements and conditions that must be observed to ensure compliance with current FCC/IC RF exposure guidelines.
- A label must be affixed to the outside of the host product with the following statements:
  This device contains FCC ID: R7T1001102
  This equipment contains equipment certified under IC ID: 5136A-1001102
- 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.
- The final host / module combination may also need to be evaluated against the FCC Part 15C criteria for intentional radiators according KDB 996369.
- 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.

OEM requirements:
The OEM must ensure that the following conditions are met.

- 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: R7T1001102
  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 ensure that timing requirements according to 47 CFR 15.231(a-c) are met.
  - The module must be used with only the following approved antenna(s).

27.7 Pre-certified antennas

The Cordelia-I is pre-certified with the following antennas.

Product	Certified antenna
2610011025000	PCB antenna included in the Cordelia-I
2610011025000	Dipole antenna as specified in chapter 20.3.4.1

It is only possible to connect an antenna by soldering. It is mandatory to follow chapter 21.3 when connecting an antenna. Special care must be taken when using an external antenna to fullfil the requirement of permanently attached antenna or unique coupling for example by using the certified dipole antenna in a closed housing, so that only through professional installation it is possible to remove it.

28 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 28 and 28 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*.

29 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.

30 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.

31 Error codes

The section briefly describes the meaning of error codes returned by Cordelia-I in response to commands.

31.1 AT command parse errors

```
STRMPL_ERROR_PARAM_MISSING      (-1)
STRMPL_ERROR_MEM_ALLOCATION     (-2)
STRMPL_ERROR_DELIM_MISSING      (-3)
STRMPL_ERROR_WRONG_PARAM        (-4)
STRMPL_ERROR_WRONG_SIZE         (-5)
```

31.2 Disconnection reason codes

```
/* WLAN Disconnect Reason Codes */
 SL_WLAN_DISCONNECT_UNSPECIFIED (1)
 SL_WLAN_DISCONNECT_AUTH_NO_LONGER_VALID (2)
 SL_WLAN_DISCONNECT_DEAUTH_SENDING_STA_LEAVING (3)
 SL_WLAN_DISCONNECT_INACTIVITY (4)
 SL_WLAN_DISCONNECT_TOO_MANY_STA (5)
 SL_WLAN_DISCONNECT_FRAME_FROM_NONAUTH_STA (6)
 SL_WLAN_DISCONNECT_FRAME_FROM_NONASSOC_STA (7)
 SL_WLAN_DISCONNECT_DISS_SENDING_STA_LEAVING (8)
 SL_WLAN_DISCONNECT_STA_NOT_AUTH (9)
 SL_WLAN_DISCONNECT_POWER_CAPABILITY_INVALID (10)
 SL_WLAN_DISCONNECT_SUPPORTED_CHANNELS_INVALID (11)
 SL_WLAN_DISCONNECT_INVALID_IE (13)
 SL_WLAN_DISCONNECT_MIC_FAILURE (14)
 SL_WLAN_DISCONNECT_FOURWAY_HANDSHAKE_TIMEOUT (15)
 SL_WLAN_DISCONNECT_GROUPKEY_HANDSHAKE_TIMEOUT (16)
 SL_WLAN_DISCONNECT_REASSOC_INVALID_IE (17)
 SL_WLAN_DISCONNECT_INVALID_GROUP_CIPHER (18)
 SL_WLAN_DISCONNECT_INVALID_PAIRWISE_CIPHER (19)
 SL_WLAN_DISCONNECT_INVALID_AKMP (20)
 SL_WLAN_DISCONNECT_UNSUPPORTED_RSN_VERSION (21)
 SL_WLAN_DISCONNECT_INVALID_RSN_CAPABILITIES (22)
 SL_WLAN_DISCONNECT_IEEE_802_1X_AUTHENTICATION_FAILED (23)
 SL_WLAN_DISCONNECT_CIPHER_SUITE_REJECTED (24)
 SL_WLAN_DISCONNECT_DISASSOC_QOS (32)
 SL_WLAN_DISCONNECT_DISASSOC_QOS_BANDWIDTH (33)
 SL_WLAN_DISCONNECT_DISASSOC_EXCESSIVE_ACK_PENDING (34)
 SL_WLAN_DISCONNECT_DISASSOC_TXOP_LIMIT (35)
 SL_WLAN_DISCONNECT_STA_LEAVING (36)
 SL_WLAN_DISCONNECT_STA_DECLINED (37)
 SL_WLAN_DISCONNECT_STA_UNKNOWN_BA (38)
 SL_WLAN_DISCONNECT_STA_TIMEOUT (39)
 SL_WLAN_DISCONNECT_STA_UNSUPPORTED_CIPHER_SUITE (40)
 SL_WLAN_DISCONNECT_USER_INITIATED (200)
 SL_WLAN_DISCONNECT_AUTH_TIMEOUT (202)
 SL_WLAN_DISCONNECT_ASSOC_TIMEOUT (203)
 SL_WLAN_DISCONNECT_SECURITY_FAILURE (204)
 SL_WLAN_DISCONNECT_WHILE_CONNNECTING (208)
```

```
SL_WLAN_DISCONNECT_MISSING_CERT (209)
SL_WLAN_DISCONNECT_CERTIFICATE_EXPIRED (210)
```

31.3 Socket error codes

```
/* BSD SOCKET ERRORS CODES */

SL_ERROR_BSD_SOC_ERROR (-1L) /* Failure */
SL_ERROR_BSD_EINTR (-4L) /* Interrupted system call */
SL_ERROR_BSD_E2BIG (-7L) /* length too big */
SL_ERROR_BSD_INEXE (-8L) /* socket command in execution */
SL_ERROR_BSD_EBADF (-9L) /* Bad file number */
SL_ERROR_BSD_ENSOCK (-10L) /* The system limit on the total number of open socket, has been
    reached */
SL_ERROR_BSD_EAGAIN (-11L) /* Try again */
SL_ERROR_BSD_EWOULDBLOCK SL_ERROR_BSD_EAGAIN
SL_ERROR_BSD_ENOMEM (-12L) /* Out of memory */
SL_ERROR_BSD_EACCES (-13L) /* Permission denied */
SL_ERROR_BSD_EFAULT (-14L) /* Bad address */
SL_ERROR_BSD_ECLOSE (-15L) /* close socket operation failed to transmit all queued packets */
SL_ERROR_BSD_EALREADY_ENABLED (-21L) /* Transceiver - Transceiver already ON. there could be
    only one */
SL_ERROR_BSD_EINVAL (-22L) /* Invalid argument */
SL_ERROR_BSD_EAUTO_CONNECT_OR_CONNECTING (-69L) /* Transceiver - During connection, connected
    or auto mode started */
SL_ERROR_BSD_CONNECTION_PENDING (-72L) /* Transceiver - Device is connected, disconnect first
    to open transceiver */
SL_ERROR_BSD_EUNSUPPORTED_ROLE (-86L) /* Transceiver - Trying to start when WLAN role is AP or
     P2P GO */
SL_ERROR_BSD_EDESTADDRREQ (-89L) /* Destination address required */
SL_ERROR_BSD_EPROTOTYPE (-91L) /* Protocol wrong type for socket */
SL_ERROR_BSD_ENOPROTOOPT (-92L) /* Protocol not available */
SL_ERROR_BSD_EPROTONOSUPPORT (-93L) /* Protocol not supported */
SL_ERROR_BSD_ESOCKTNOSUPPORT (-94L) /* Socket type not supported */
SL_ERROR_BSD_EOPNOTSUPP (-95L) /* Operation not supported on transport endpoint */
SL_ERROR_BSD_EAFNOSUPPORT (-97L) /* Address family not supported by protocol */
SL_ERROR_BSD_EADDRINUSE (-98L) /* Address already in use */
SL_ERROR_BSD_EADDRNOTAVAIL (-99L) /* Cannot assign requested address */
SL_ERROR_BSD_ENETUNREACH (-101L) /* Network is unreachable */
SL_ERROR_BSD_ENOBUFS (-105L) /* No buffer space available */
SL_ERROR_BSD_EOBUFF SL_ENOBUFS
SL_ERROR_BSD_EISCONN (-106L) /* Transport endpoint is already connected */
SL_ERROR_BSD_ENOTCONN (-107L) /* Transport endpoint is not connected */
SL_ERROR_BSD_ETIMEDOUT (-110L) /* Connection timed out */
SL_ERROR_BSD_ECONNREFUSED (-111L) /* Connection refused */
SL_ERROR_BSD_EALREADY (-114L) /* Non blocking connect in progress, try again */
```

31.4 Secure socket error codes

```
/* ssl tls security start with -300 offset */
SL_ERROR_BSD_ESEC_CLOSE_NOTIFY (-300L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_UNEXPECTED_MESSAGE (-310L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_BAD_RECORD_MAC (-320L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_DECRYPTION_FAILED (-321L) /* ssl/tls alerts */
```

```
SL_ERROR_BSD_ESEC_RECORD_OVERFLOW (-322L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_DECOMPRESSION_FAILURE (-330L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_HANDSHAKE_FAILURE (-340L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_NO_CERTIFICATE (-341L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_BAD_CERTIFICATE (-342L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_UNSUPPORTED_CERTIFICATE (-343L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_CERTIFICATE_REVOKED (-344L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_CERTIFICATE_EXPIRED (-345L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_CERTIFICATE_UNKNOWN (-346L) /* ssl/tls alerts */

SL_ERROR_BSD_ESEC_ILLEGAL_PARAMETER (-347L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_ACCESS_DENIED (-349L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_DECODE_ERROR (-350L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_DECRYPT_ERROR1 (-351L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_EXPORT_RESTRICTION (-360L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_PROTOCOL_VERSION (-370L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_INSUFFICIENT_SECURITY (-371L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_INTERNAL_ERROR (-380L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_USER_CANCELLED (-390L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_NO_RENEGOTIATION (-400L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_UNSUPPORTED_EXTENSION (-410L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_CERTIFICATE_UNOBTAINABLE (-411L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_UNRECOGNIZED_NAME (-412L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_BAD_CERTIFICATE_STATUS_RESPONSE (-413L) /* ssl/tls alerts */
SL_ERROR_BSD_ESEC_BAD_CERTIFICATE_HASH_VALUE (-414L) /* ssl/tls alerts */
/* propriety secure */
SL_ERROR_BSD_ESECGENERAL (-450L) /* error secure level general error */
SL_ERROR_BSD_ESECDECRYPT (-451L) /* error secure level, decrypt recv packet fail */
SL_ERROR_BSD_ESECCLOSED (-452L) /* secure layrer is closed by other size , tcp is still
    connected */
SL_ERROR_BSD_ESECSNOVERIFY (-453L) /* Connected without server verification */
SL_ERROR_BSD_ESECNOCAFILE (-454L) /* error secure level CA file not found*/
SL_ERROR_BSD_ESECMEMORY (-455L) /* error secure level No memory space available */
SL_ERROR_BSD_ESECBADCAFILE (-456L) /* error secure level bad CA file */
SL_ERROR_BSD_ESECBADCERTFILE (-457L) /* error secure level bad Certificate file */
SL_ERROR_BSD_ESECBADPRIVATEFILE (-458L) /* error secure level bad private file */
SL_ERROR_BSD_ESECBADDHFILE (-459L) /* error secure level bad DH file */
SL_ERROR_BSD_ESECTOOMANYSSLOPENED (-460L) /* MAX SSL Sockets are opened */
SL_ERROR_BSD_ESECDATEERROR (-461L) /* connected with certificate date verification error */
SL_ERROR_BSD_ESECHANDSHAKETIMEDOUT (-462L) /* connection timed out due to handshake time */
SL_ERROR_BSD_ESECTXBUFFERNOTEMPTY (-463L) /* cannot start ssl connection while send buffer is
    full */
SL_ERROR_BSD_ESECRXBUFFERNOTEMPTY (-464L) /* cannot start ssl connection while recv buffer is
    full */
SL_ERROR_BSD_ESECSSLDURINGHANDSHAKE (-465L) /* cannot use while in hanshaking */
SL_ERROR_BSD_ESECNOTALLOWEDWHENLISTENING (-466L) /* the operation is not allowed when
    listening, do before listen*/
SL_ERROR_BSD_ESECCERTIFICATEREVOKED (-467L) /* connected but on of the certificates in the
    chain is revoked */
SL_ERROR_BSD_ESECUNKNOWNROOTCA (-468L) /* connected but the root CA used to validate the peer
    is unknown */
SL_ERROR_BSD_ESECWRONGPEERCERT (-469L) /* wrong peer cert (server cert) was received while
    trying to connect to server */
SL_ERROR_BSD_ESECTCPDISCONNECTEDUNCOMPLETERECORD (-470L) /* the other side disconnected the
    TCP layer and didn't send the whole ssl record */

SL_ERROR_BSD_ESEC_BUFFER_E (-632L) /* output buffer too small or input too large */
SL_ERROR_BSD_ESEC_ALGO_ID_E (-633L) /* setting algo id error */
```

```
SL_ERROR_BSD_ESEC_PUBLIC_KEY_E (-634L) /* setting public key error */
SL_ERROR_BSD_ESEC_DATE_E (-635L) /* setting date validity error */
SL_ERROR_BSD_ESEC_SUBJECT_E (-636L) /* setting subject name error */
SL_ERROR_BSD_ESEC_ISSUER_E (-637L) /* setting issuer name error */
SL_ERROR_BSD_ESEC_CA_TRUE_E (-638L) /* setting CA basic constraint true error */
SL_ERROR_BSD_ESEC_EXTENSIONS_E (-639L) /* setting extensions error */
SL_ERROR_BSD_ESEC_ASN_PARSE_E (-640L) /* ASN parsing error, invalid input */
SL_ERROR_BSD_ESEC_ASN_VERSION_E (-641L) /* ASN version error, invalid number */
SL_ERROR_BSD_ESEC_ASN_GETINT_E (-642L) /* ASN get big int error, invalid data */
SL_ERROR_BSD_ESEC_ASN_RSA_KEY_E (-643L) /* ASN key init error, invalid input */
SL_ERROR_BSD_ESEC_ASN_OBJECT_ID_E (-644L) /* ASN object id error, invalid id */
SL_ERROR_BSD_ESEC_ASN_TAG_NULL_E (-645L) /* ASN tag error, not null */
SL_ERROR_BSD_ESEC_ASN_EXPECT_0_E (-646L) /* ASN expect error, not zero */
SL_ERROR_BSD_ESEC_ASN_BITSTR_E (-647L) /* ASN bit string error, wrong id */
SL_ERROR_BSD_ESEC_ASN_UNKNOWN_OID_E (-648L) /* ASN oid error, unknown sum id */
SL_ERROR_BSD_ESEC_ASN_DATE_SZ_E (-649L) /* ASN date error, bad size */
SL_ERROR_BSD_ESEC_ASN_BEFORE_DATE_E (-650L) /* ASN date error, current date before */
SL_ERROR_BSD_ESEC_ASN_AFTER_DATE_E (-651L) /* ASN date error, current date after */
SL_ERROR_BSD_ESEC_ASN_SIG_OID_E (-652L) /* ASN signature error, mismatched oid */
SL_ERROR_BSD_ESEC_ASN_TIME_E (-653L) /* ASN time error, unknown time type */
SL_ERROR_BSD_ESEC_ASN_INPUT_E (-654L) /* ASN input error, not enough data */
SL_ERROR_BSD_ESEC_ASN_SIG_CONFIRM_E (-655L) /* ASN sig error, confirm failure */
SL_ERROR_BSD_ESEC_ASN_SIG_HASH_E (-656L) /* ASN sig error, unsupported hash type */
SL_ERROR_BSD_ESEC_ASN_SIG_KEY_E (-657L) /* ASN sig error, unsupported key type */
SL_ERROR_BSD_ESEC_ASN_DH_KEY_E (-658L) /* ASN key init error, invalid input */
SL_ERROR_BSD_ESEC_ASN_NTRU_KEY_E (-659L) /* ASN ntru key decode error, invalid input */
SL_ERROR_BSD_ESEC_ASN_CRIT_EXT_E (-660L) /* ASN unsupported critical extension */
SL_ERROR_BSD_ESEC_ECC_BAD_ARG_E (-670L) /* ECC input argument of wrong type */
SL_ERROR_BSD_ESEC_ASN_ECC_KEY_E (-671L) /* ASN ECC bad input */
SL_ERROR_BSD_ESEC_ECC_CURVE_OID_E (-672L) /* Unsupported ECC OID curve type */
SL_ERROR_BSD_ESEC_BAD_FUNC_ARG (-673L) /* Bad function argument provided */
SL_ERROR_BSD_ESEC_NOT_COMPILED_IN (-674L) /* Feature not compiled in */
SL_ERROR_BSD_ESEC_UNICODE_SIZE_E (-675L) /* Unicode password too big */
SL_ERROR_BSD_ESEC_NO_PASSWORD (-676L) /* no password provided by user */
SL_ERROR_BSD_ESEC_ALT_NAME_E (-677L) /* alt name size problem, too big */
SL_ERROR_BSD_ESEC_ASN_NO_SIGNER_E (-688L) /* ASN no signer to confirm failure */
SL_ERROR_BSD_ESEC_ASN_CRL_CONFIRM_E (-689L) /* ASN CRL signature confirm failure */
SL_ERROR_BSD_ESEC_ASN_CRL_NO_SIGNER_E (-690L) /* ASN CRL no signer to confirm failure */
SL_ERROR_BSD_ESEC_ASN_OCSP_CONFIRM_E (-691L) /* ASN OCSP signature confirm failure */
SL_ERROR_BSD_ESEC_VERIFY_FINISHED_ERROR (-704L) /* verify problem on finished */
SL_ERROR_BSD_ESEC_VERIFY_MAC_ERROR (-705L) /* verify mac problem */
SL_ERROR_BSD_ESEC_PARSE_ERROR (-706L) /* parse error on header */
SL_ERROR_BSD_ESEC_UNKNOWN_HANDSHAKE_TYPE (-707L) /* weird handshake type */
SL_ERROR_BSD_ESEC_SOCKET_ERROR_E (-708L) /* error state on socket */
SL_ERROR_BSD_ESEC_SOCKET_NODATA (-709L) /* expected data, not there */
SL_ERROR_BSD_ESEC_INCOMPLETE_DATA (-710L) /* don't have enough data to complete task */
SL_ERROR_BSD_ESEC_UNKNOWN_RECORD_TYPE (-711L) /* unknown type in record hdr */
SL_ERROR_BSD_ESEC_INNER_DECRYPT_ERROR (-712L) /* error during decryption */
SL_ERROR_BSD_ESEC_FATAL_ERROR (-713L) /* recvd alert fatal error */
SL_ERROR_BSD_ESEC_ENCRYPT_ERROR (-714L) /* error during encryption */
SL_ERROR_BSD_ESEC_FREAD_ERROR (-715L) /* fread problem */
SL_ERROR_BSD_ESEC_NO_PEER_KEY (-716L) /* need peer's key */
SL_ERROR_BSD_ESEC_NO_PRIVATE_KEY (-717L) /* need the private key */
SL_ERROR_BSD_ESEC_RSA_PRIVATE_ERROR (-718L) /* error during rsa priv op */
SL_ERROR_BSD_ESEC_NO_DH_PARAMS (-719L) /* server missing DH params */
SL_ERROR_BSD_ESEC_BUILD_MSG_ERROR (-720L) /* build message failure */
SL_ERROR_BSD_ESEC_BAD_HELLO (-721L) /* client hello malformed */
SL_ERROR_BSD_ESEC_DOMAIN_NAME_MISMATCH (-722L) /* peer subject name mismatch */
```

```
SL_ERROR_BSD_ESEC_WANT_READ (-723L) /* want read, call again */
SL_ERROR_BSD_ESEC_NOT_READY_ERROR (-724L) /* handshake layer not ready */
SL_ERROR_BSD_ESEC_PMS_VERSION_ERROR (-725L) /* pre m secret version error */
SL_ERROR_BSD_ESEC_WANT_WRITE (-727L) /* want write, call again */
SL_ERROR_BSD_ESEC_BUFFER_ERROR (-728L) /* malformed buffer input */
SL_ERROR_BSD_ESEC_VERIFY_CERT_ERROR (-729L) /* verify cert error */
SL_ERROR_BSD_ESEC_VERIFY_SIGN_ERROR (-730L) /* verify sign error */
SL_ERROR_BSD_ESEC_LENGTH_ERROR (-741L) /* record layer length error */
SL_ERROR_BSD_ESEC_PEER_KEY_ERROR (-742L) /* can't decode peer key */
SL_ERROR_BSD_ESEC_ZERO_RETURN (-743L) /* peer sent close notify */
SL_ERROR_BSD_ESEC_SIDE_ERROR (-744L) /* wrong client/server type */
SL_ERROR_BSD_ESEC_NO_PEER_CERT (-745L) /* peer didn't send key */
SL_ERROR_BSD_ESEC_ECC_CURVETYPE_ERROR (-750L) /* Bad ECC Curve Type */
SL_ERROR_BSD_ESEC_ECC_CURVE_ERROR (-751L) /* Bad ECC Curve */
SL_ERROR_BSD_ESEC_ECC_PEERKEY_ERROR (-752L) /* Bad Peer ECC Key */
SL_ERROR_BSD_ESEC_ECC_MAKEKEY_ERROR (-753L) /* Bad Make ECC Key */
SL_ERROR_BSD_ESEC_ECC_EXPORT_ERROR (-754L) /* Bad ECC Export Key */
SL_ERROR_BSD_ESEC_ECC_SHARED_ERROR (-755L) /* Bad ECC Shared Secret */
SL_ERROR_BSD_ESEC_NOT_CA_ERROR (-757L) /* Not a CA cert error */
SL_ERROR_BSD_ESEC_BAD_PATH_ERROR (-758L) /* Bad path for opendir */
SL_ERROR_BSD_ESEC_BAD_CERT_MANAGER_ERROR (-759L) /* Bad Cert Manager */
SL_ERROR_BSD_ESEC_OCSP_CERT_REVOKED (-760L) /* OCSP Certificate revoked */
SL_ERROR_BSD_ESEC_CRL_CERT_REVOKED (-761L) /* CRL Certificate revoked */
SL_ERROR_BSD_ESEC_CRL_MISSING (-762L) /* CRL Not loaded */
SL_ERROR_BSD_ESEC_MONITOR_RUNNING_E (-763L) /* CRL Monitor already running */
SL_ERROR_BSD_ESEC_THREAD_CREATE_E (-764L) /* Thread Create Error */
SL_ERROR_BSD_ESEC_OCSP_NEED_URL (-765L) /* OCSP need an URL for lookup */
SL_ERROR_BSD_ESEC_OCSP_CERT_UNKNOWN (-766L) /* OCSP responder doesn't know */
SL_ERROR_BSD_ESEC_OCSP_LOOKUP_FAIL (-767L) /* OCSP lookup not successful */
SL_ERROR_BSD_ESEC_MAX_CHAIN_ERROR (-768L) /* max chain depth exceeded */
SL_ERROR_BSD_ESEC_NO_PEER_VERIFY (-778L) /* Need peer cert verify Error */
SL_ERROR_BSD_ESEC_UNSUPPORTED_SUITE (-790L) /* unsupported cipher suite */
SL_ERROR_BSD_ESEC_MATCH_SUITE_ERROR (-791L) /* can't match cipher suite */
```

31.5 WLAN error codes

```
/* WLAN ERRORS CODES*/
 SL_ERROR_WLAN_KEY_ERROR (-2049L)
 SL_ERROR_WLAN_INVALID_ROLE (-2050L)
 SL_ERROR_WLAN_PREFERRED_NETWORKS_FILE_LOAD_FAILED (-2051L)
 SL_ERROR_WLAN_CANNOT_CONFIG_SCAN_DURING_PROVISIONING (-2052L)
 SL_ERROR_WLAN_INVALID_SECURITY_TYPE (-2054L)
 SL_ERROR_WLAN_PASSPHRASE_TOO_LONG (-2055L)
 SL_ERROR_WLAN_EAP_WRONG_METHOD (-2057L)
 SL_ERROR_WLAN_PASSWORD_ERROR (-2058L)
 SL_ERROR_WLAN_EAP_ANONYMOUS_LEN_ERROR (-2059L)
 SL_ERROR_WLAN_SSID_LEN_ERROR (-2060L)
 SL_ERROR_WLAN_USER_ID_LEN_ERROR (-2061L)
 SL_ERROR_WLAN_PREFERRED_NETWORK_LIST_FULL (-2062L)
 SL_ERROR_WLAN_PREFERRED_NETWORKS_FILE_WRITE_FAILED (-2063L)
 SL_ERROR_WLAN_ILLEGAL_WEP_KEY_INDEX (-2064L)
 SL_ERROR_WLAN_INVALID_DWELL_TIME_VALUES (-2065L)
 SL_ERROR_WLAN_INVALID_POLICY_TYPE (-2066L)
 SL_ERROR_WLAN_PM_POLICY_INVALID_OPTION (-2067L)
 SL_ERROR_WLAN_PM_POLICY_INVALID_PARAMS (-2068L)
 SL_ERROR_WLAN_WIFI_NOT_CONNECTED (-2069L)
```

```
 SL_ERROR_WLAN_ILLEGAL_CHANNEL (-2070L)
 SL_ERROR_WLAN_WIFI_ALREADY_DISCONNECTED (-2071L)
 SL_ERROR_WLAN_TRANSCEIVER_ENABLED (-2072L)
 SL_ERROR_WLAN_GET_NETWORK_LIST_EAGAIN (-2073L)
 SL_ERROR_WLAN_GET_PROFILE_INVALID_INDEX (-2074L)
 SL_ERROR_WLAN_FAST_CONN_DATA_INVALID (-2075L)
 SL_ERROR_WLAN_NO_FREE_PROFILE (-2076L)
 SL_ERROR_WLAN_AP_SCAN_INTERVAL_TOO_LOW (-2077L)
 SL_ERROR_WLAN_SCAN_POLICY_INVALID_PARAMS (-2078L)
SL_ERROR_WLAN_INVALID_COUNTRY_CODE (-2164L)
SL_ERROR_WLAN_NVMEM_ACCESS_FAILED (-2165L)
SL_ERROR_WLAN_OLD_FILE_VERSION (-2166L)
SL_ERROR_WLAN_TX_POWER_OUT_OF_RANGE (-2167L)
SL_ERROR_WLAN_INVALID_AP_PASSWORD_LENGTH (-2168L)

SL_ERROR_WLAN_PROVISIONING_ABORT_PROVISIONING_ALREADY_STARTED (-2169L)
SL_ERROR_WLAN_PROVISIONING_ABORT_HTTP_SERVER_DISABLED (-2170L)
SL_ERROR_WLAN_PROVISIONING_ABORT_PROFILE_LIST_FULL (-2171L)
SL_ERROR_WLAN_PROVISIONING_ABORT_INVALID_PARAM (-2172L)
SL_ERROR_WLAN_PROVISIONING_ABORT_GENERAL_ERROR (-2173L)
SL_ERROR_WLAN_MULTICAST_EXCEED_MAX_ADDR (-2174L)
SL_ERROR_WLAN_MULTICAST_INVAL_ADDR (-2175L)
SL_ERROR_WLAN_AP_SCAN_INTERVAL_TOO_SHORT (-2176L)
SL_ERROR_WLAN_PROVISIONING_CMD_NOT_EXPECTED (-2177L)


SL_ERROR_WLAN_AP_ACCESS_LIST_NO_ADDRESS_TO_DELETE (-2178L) /* List is empty, no address to
    delete */
SL_ERROR_WLAN_AP_ACCESS_LIST_FULL (-2179L) /* access list is full */
SL_ERROR_WLAN_AP_ACCESS_LIST_DISABLED (-2180L) /* access list is disabled */
SL_ERROR_WLAN_AP_ACCESS_LIST_MODE_NOT_SUPPORTED (-2181L) /* Trying to switch to unsupported
    mode */
SL_ERROR_WLAN_AP_STA_NOT_FOUND (-2182L) /* trying to disconnect station which is not connected
     */
```

31.6 Device error codes

```
/* DEVICE ERRORS CODES*/
SL_ERROR_SUPPLICANT_ERROR (-4097L)
SL_ERROR_HOSTAPD_INIT_FAIL (-4098L)
SL_ERROR_HOSTAPD_INIT_IF_FAIL (-4099L)
SL_ERROR_WLAN_DRV_INIT_FAIL (-4100L)
SL_ERROR_FS_FILE_TABLE_LOAD_FAILED (-4102L) /* init file system failed */
SL_ERROR_MDNS_ENABLE_FAIL (-4103L) /* mDNS enable failed */
SL_ERROR_ROLE_STA_ERR (-4107L) /* Failure to load MAC/PHY in STA role */
SL_ERROR_ROLE_AP_ERR (-4108L) /* Failure to load MAC/PHY in AP role */
SL_ERROR_ROLE_P2P_ERR (-4109L) /* Failure to load MAC/PHY in P2P role */
SL_ERROR_CALIB_FAIL (-4110L) /* Failure of calibration */
SL_ERROR_FS_CORRUPTED_ERR (-4111L) /* FS is corrupted, Return to Factory Image or Program new
    image should be invoked (see sl_FsCtl, sl_FsProgram) */
SL_ERROR_FS_ALERT_ERR (-4112L) /* Device is locked, Return to Factory Image or Program new
    image should be invoked (see sl_FsCtl, sl_FsProgram) */
SL_ERROR_RESTORE_IMAGE_COMPLETE (-4113L) /* Return to factory image completed, perform reset
    */
SL_ERROR_UNKNOWN_ERR (-4114L)
SL_ERROR_GENERAL_ERR (-4115L) /* General error during init */
```

```
SL_ERROR_WRONG_ROLE (-4116L)
SL_ERROR_INCOMPLETE_PROGRAMMING (-4117L) /* Error during programming, Program new image should
     be invoked (see sl_FsProgram) */


SL_ERROR_PENDING_TXRX_STOP_TIMEOUT_EXP (-4118L) /* Timeout expired before completing all TX\RX
     */
SL_ERROR_PENDING_TXRX_NO_TIMEOUT (-4119L) /* No Timeout , still have pending TX\RX */
SL_ERROR_INVALID_PERSISTENT_CONFIGURATION (-4120L) /* persistency configuration can only be
    set to 0 (disabled) or 1 (enabled) */
```

31.7 Network config error codes

```
/* NETCFG ERRORS CODES*/
SL_ERROR_STATIC_ADDR_SUBNET_ERROR (-8193L)
SL_ERROR_INCORRECT_IPV6_STATIC_LOCAL_ADDR (-8194L) /* Ipv6 Local address perfix is wrong */
SL_ERROR_INCORRECT_IPV6_STATIC_GLOBAL_ADDR (-8195L) /* Ipv6 Global address perfix is wrong */
SL_ERROR_IPV6_LOCAL_ADDR_SHOULD_BE_SET_FIRST (-8196L) /* Attempt to set ipv6 global address
    before ipv6 local address is set */
```

31.8 File System error codes

```
/* FS ERRORS CODES*/
SL_FS_OK (0L)
SL_ERROR_FS_EXTRACTION_WILL_START_AFTER_RESET (-10241L)
SL_ERROR_FS_NO_CERTIFICATE_STORE (-10242L)
SL_ERROR_FS_IMAGE_SHOULD_BE_AUTHENTICATE (-10243L)
SL_ERROR_FS_IMAGE_SHOULD_BE_ENCRYPTED (-10244L)
SL_ERROR_FS_IMAGE_CANT_BE_ENCRYPTED (-10245L)
SL_ERROR_FS_DEVELOPMENT_BOARD_WRONG_MAC (-10246L)
SL_ERROR_FS_DEVICE_NOT_SECURED (-10247L)
SL_ERROR_FS_SYSTEM_FILE_ACCESS_DENIED (-10248L)
SL_ERROR_FS_IMAGE_EXTRACT_EXPECTING_USER_KEY (-10249L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_CLOSE_FILE (-10250L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_WRITE_FILE (-10251L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_OPEN_FILE (-10252L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_GET_IMAGE_HEADER (-10253L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_GET_IMAGE_INFO (-10254L)
SL_ERROR_FS_IMAGE_EXTRACT_SET_ID_NOT_EXIST (-10255L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_DELETE_FILE (-10256L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_FORMAT_FS (-10257L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_LOAD_FS (-10258L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_GET_DEV_INFO (-10259L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_DELETE_STORAGE (-10260L)
SL_ERROR_FS_IMAGE_EXTRACT_INCORRECT_IMAGE_LOCATION (-10261L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_CREATE_IMAGE_FILE (-10262L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_INIT (-10263L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_LOAD_FILE_TABLE (-10264L)
SL_ERROR_FS_IMAGE_EXTRACT_ILLEGAL_COMMAND (-10266L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_WRITE_FAT (-10267L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_RET_FACTORY_DEFAULT (-10268L)
SL_ERROR_FS_IMAGE_EXTRACT_FAILED_TO_READ_NV (-10269L)
SL_ERROR_FS_PROGRAMMING_IMAGE_NOT_EXISTS (-10270L)
```

```
SL_ERROR_FS_PROGRAMMING_IN_PROCESS (-10271L)
SL_ERROR_FS_PROGRAMMING_ALREADY_STARTED (-10272L)
SL_ERROR_FS_CERT_IN_THE_CHAIN_REVOKED_SECURITY_ALERT (-10273L)
SL_ERROR_FS_INIT_CERTIFICATE_STORE (-10274L)
SL_ERROR_FS_PROGRAMMING_ILLEGAL_FILE (-10275L)
SL_ERROR_FS_PROGRAMMING_NOT_STARTED (-10276L)
SL_ERROR_FS_IMAGE_EXTRACT_NO_FILE_SYSTEM (-10277L)
SL_ERROR_FS_WRONG_INPUT_SIZE (-10278L)
SL_ERROR_FS_BUNDLE_FILE_SHOULD_BE_CREATED_WITH_FAILSAFE (-10279L)
SL_ERROR_FS_BUNDLE_NOT_CONTAIN_FILES (-10280L)
SL_ERROR_FS_BUNDLE_ALREADY_IN_STATE (-10281L)
SL_ERROR_FS_BUNDLE_NOT_IN_CORRECT_STATE (-10282L)
SL_ERROR_FS_BUNDLE_FILES_ARE_OPENED (-10283L)
SL_ERROR_FS_INCORRECT_FILE_STATE_FOR_OPERATION (-10284L)
SL_ERROR_FS_EMPTY_SFLASH (-10285L)
SL_ERROR_FS_FILE_IS_NOT_SECURE_AND_SIGN (-10286L)
SL_ERROR_FS_ROOT_CA_IS_UNKOWN (-10287L)
SL_ERROR_FS_FILE_HAS_NOT_BEEN_CLOSE_CORRECTLY (-10288L)
SL_ERROR_FS_WRONG_SIGNATURE_SECURITY_ALERT (-10289L)
SL_ERROR_FS_WRONG_SIGNATURE_OR_CERTIFIC_NAME_LENGTH (-10290L)
SL_ERROR_FS_NOT_16_ALIGNED (-10291L)
SL_ERROR_FS_CERT_CHAIN_ERROR_SECURITY_ALERT (-10292L)
SL_ERROR_FS_FILE_NAME_EXIST (-10293L)
SL_ERROR_FS_EXTENDED_BUF_ALREADY_ALLOC (-10294L)
SL_ERROR_FS_FILE_SYSTEM_NOT_SECURED (-10295L)
SL_ERROR_FS_OFFSET_NOT_16_BYTE_ALIGN (-10296L)
SL_ERROR_FS_FAILED_READ_NVMEM (-10297L)
SL_ERROR_FS_WRONG_FILE_NAME (-10298L)
SL_ERROR_FS_FILE_SYSTEM_IS_LOCKED (-10299L)
SL_ERROR_FS_SECURITY_ALERT (-10300L)
SL_ERROR_FS_FILE_INVALID_FILE_SIZE (-10301L)
SL_ERROR_FS_INVALID_TOKEN (-10302L)
SL_ERROR_FS_NO_DEVICE_IS_LOADED (-10303L)
SL_ERROR_FS_SECURE_CONTENT_INTEGRITY_FAILURE (-10304L)
SL_ERROR_FS_SECURE_CONTENT_RETRIVE_ASYMETRIC_KEY_ERROR (-10305L)
SL_ERROR_FS_OVERLAP_DETECTION_THRESHHOLD (-10306L)
SL_ERROR_FS_FILE_HAS_RESERVED_NV_INDEX (-10307L)
SL_ERROR_FS_FILE_MAX_SIZE_EXCEEDED (-10310L)
SL_ERROR_FS_INVALID_READ_BUFFER (-10311L)
SL_ERROR_FS_INVALID_WRITE_BUFFER (-10312L)
SL_ERROR_FS_FILE_IMAGE_IS_CORRUPTED (-10313L)
SL_ERROR_FS_SIZE_OF_FILE_EXT_EXCEEDED (-10314L)
SL_ERROR_FS_WARNING_FILE_NAME_NOT_KEPT (-10315L)
SL_ERROR_FS_MAX_OPENED_FILE_EXCEEDED (-10316L)
SL_ERROR_FS_FAILED_WRITE_NVMEM_HEADER (-10317L)
SL_ERROR_FS_NO_AVAILABLE_NV_INDEX (-10318L)
SL_ERROR_FS_FAILED_TO_ALLOCATE_MEM (-10319L)
SL_ERROR_FS_OPERATION_BLOCKED_BY_VENDOR (-10320L)
SL_ERROR_FS_FAILED_TO_READ_NVMEM_FILE_SYSTEM (-10321L)
SL_ERROR_FS_NOT_ENOUGH_STORAGE_SPACE (-10322L)
SL_ERROR_FS_INIT_WAS_NOT_CALLED (-10323L)
SL_ERROR_FS_FILE_SYSTEM_IS_BUSY (-10324L)
SL_ERROR_FS_INVALID_ACCESS_TYPE (-10325L)
SL_ERROR_FS_FILE_ALREADY_EXISTS (-10326L)
SL_ERROR_FS_PROGRAM_FAILURE (-10327L)
SL_ERROR_FS_NO_ENTRIES_AVAILABLE (-10328L)
SL_ERROR_FS_FILE_ACCESS_IS_DIFFERENT (-10329L)
SL_ERROR_FS_INVALID_FILE_MODE (-10330L)
```

```
SL_ERROR_FS_FAILED_READ_NVFILE (-10331L)
SL_ERROR_FS_FAILED_INIT_STORAGE (-10332L)
SL_ERROR_FS_FILE_HAS_NO_FAILSAFE (-10333L)
SL_ERROR_FS_NO_VALID_COPY_EXISTS (-10334L)
SL_ERROR_FS_INVALID_HANDLE (-10335L)
SL_ERROR_FS_FAILED_TO_WRITE (-10336L)
SL_ERROR_FS_OFFSET_OUT_OF_RANGE (-10337L)
SL_ERROR_FS_NO_MEMORY (-10338L)
SL_ERROR_FS_INVALID_LENGTH_FOR_READ (-10339L)
SL_ERROR_FS_WRONG_FILE_OPEN_FLAGS (-10340L)
SL_ERROR_FS_FILE_NOT_EXISTS (-10341L)
SL_ERROR_FS_IGNORE_COMMIT_ROLLBAC_FLAG (-10342L) /* commit rollback flag is not supported upon
    creation */
SL_ERROR_FS_INVALID_ARGS (-10343L)
SL_ERROR_FS_FILE_IS_PENDING_COMMIT (-10344L)
SL_ERROR_FS_SECURE_CONTENT_SESSION_ALREADY_EXIST (-10345L)
SL_ERROR_FS_UNKNOWN (-10346L)
SL_ERROR_FS_FILE_NAME_RESERVED (-10347L)
SL_ERROR_FS_NO_FILE_SYSTEM (-10348L)
SL_ERROR_FS_INVALID_MAGIC_NUM (-10349L)
SL_ERROR_FS_FAILED_TO_READ_NVMEM (-10350L)
SL_ERROR_FS_NOT_SUPPORTED (-10351L)
SL_ERROR_FS_JTAG_IS_OPENED_NO_FORMAT_TO_PRDUCTION (-10352L)
SL_ERROR_FS_CONFIG_FILE_RET_READ_FAILED (-10353L)
SL_ERROR_FS_CONFIG_FILE_CHECSUM_ERROR_SECURITY_ALERT (-10354L)
SL_ERROR_FS_CONFIG_FILE_NO_SUCH_FILE (-10355L)
SL_ERROR_FS_CONFIG_FILE_MEMORY_ALLOCATION_FAILED (-10356L)
SL_ERROR_FS_IMAGE_HEADER_READ_FAILED (-10357L)
SL_ERROR_FS_CERT_STORE_DOWNGRADE (-10358L)
SL_ERROR_FS_PROGRAMMING_IMAGE_NOT_VALID (-10359L)
SL_ERROR_FS_PROGRAMMING_IMAGE_NOT_VERIFIED (-10360L)
SL_ERROR_FS_RESERVE_SIZE_IS_SMALLER (-10361L)
SL_ERROR_FS_WRONG_ALLOCATION_TABLE (-10362L)
SL_ERROR_FS_ILLEGAL_SIGNATURE (-10363L)
SL_ERROR_FS_FILE_ALREADY_OPENED_IN_PENDING_STATE (-10364L)
SL_ERROR_FS_INVALID_TOKEN_SECURITY_ALERT (-10365L)
SL_ERROR_FS_NOT_SECURE (-10366L)
SL_ERROR_FS_RESET_DURING_PROGRAMMING (-10367L)
SL_ERROR_FS_CONFIG_FILE_RET_WRITE_FAILED (-10368L)
SL_ERROR_FS_FILE_IS_ALREADY_OPENED (-10369L)
SL_ERROR_FS_FILE_IS_OPEN_FOR_WRITE (-10370L)
SL_ERROR_FS_ALERT_CANT_BE_SET_ON_NON_SECURE_DEVICE (-10371L) /* Alerts can be configured on
    non-secure device. */
SL_ERROR_FS_WRONG_CERTIFICATE_FILE_NAME (-10372L)
```

31.9 HTTP Client error codes

```
/*Internal send buffer is not big enough*/
#define HTTPClient_ESENDBUFSMALL                    (-3001)

/* Buffer inserted into HTTPClient_getOpt() is not big enough.*/
HTTPClient_EGETOPTBUFSMALL                  (-3002)

/* Response received from the server is not a valid HTTP/1.1 or HTTP/1.0 response*/
HTTPClient_ERESPONSEINVALID                 (-3003)
```

```
/* Operation could not be completed. Try again.*/
HTTPClient_EINPROGRESS                      (-3004)

/* Input domain name length is too long to be read into buffer.*/
HTTPClient_EDOMAINBUFSMALL                  (-3005)

/* Allocation failed during the CB creation.*/
HTTPClient_ECBALLOCATIONFAILED              (-3006)

/* Body size is too small.*/
HTTPClient_EBODYBUFSMALL                    (-3008)

/* Invalid de-referencing a NULL pointer.*/
HTTPClient_ENULLPOINTER                     (-3009)

/* Request header allocation failed.*/
HTTPClient_EREQUESTHEADERALLOCFAILED        (-3010)

/* Request header wasn't found in the req header list.*/
HTTPClient_EREQHEADERNOTFOUND               (-3011)

/* Host request header wasn't found.*/
HTTPClient_EHOSTNOTFOUND                    (-3012)

/* Client is already connected.*/
HTTPClient_ECLIENTALREADYCONNECTED          (-3013)

/* Response is not redirectable.*/
HTTPClient_ERESPONSEISNOTREDIRECTABLE       (-3014)

/* Send couldn't be completed.*/
HTTPClient_ESENDERROR                       (-3015)

/* Location Header fields value couldn't be read completely*/
HTTPClient_EREDIRECTLOCATIONFAIL            (-3016)

/* TLS downgrade is forbidden.*/
HTTPClient_ETLSDOWNGRADEISFORBIDDEN         (-3017)

/* Wrong API parameter.*/
HTTPClient_EWRONGAPIPARAMETER               (-3018)

/* HOST already exist.*/
HTTPClient_EHOSTHEADERALREADYEXIST          (-3019)

/* Client is disconnected.*/
HTTPClient_ENOCONNECTION                    (-3020)

/* URI is not absolute*/
HTTPClient_ENOTABSOLUTEURI                  (-3021)

/* Error during creation of security attribut*/
HTTPClient_ECANTCREATESECATTRIB             (-3022)

/* General internal error*/
HTTPClient_EINTERNAL                        (-3023)

/* Buffer inserted into getHeaderByName(..) is not big enough.*/
```

```
HTTPClient_EGETCUSOMHEADERBUFSMALL          (-3024)

/* Custom response header name on getHeaderByName(..) dosn't set before.*/
HTTPClient_ENOHEADERNAMEDASINSERTED         (-3025)
```

31.10 Other error codes

```
SL_POOL_IS_EMPTY (-2000L)
SL_ESMALLBUF (-2001L)
SL_EZEROLEN (-2002L)
SL_INVALPARAM (-2003L)
SL_BAD_INTERFACE (-2004L)
SL_API_ABORTED (-2005)
SL_RET_CODE_INVALID_INPUT (-2006L)
SL_RET_CODE_SELF_ERROR (-2007L)
SL_RET_CODE_NWP_IF_ERROR (-2008L)
SL_RET_CODE_MALLOC_ERROR (-2009L)
SL_RET_CODE_PROTOCOL_ERROR (-2010L)
SL_RET_CODE_DEV_LOCKED (-2011L)
SL_RET_CODE_DEV_ALREADY_STARTED (-2012L)
SL_RET_CODE_API_COMMAND_IN_PROGRESS (-2013L)
SL_RET_CODE_PROVISIONING_IN_PROGRESS (-2014L)
SL_RET_CODE_NET_APP_PING_INVALID_PARAMS (-2015L)
SL_RET_CODE_SOCKET_SELECT_IN_PROGRESS_ERROR (-2016L)
SL_RET_CODE_STOP_IN_PROGRESS (-2017L)
SL_RET_CODE_DEV_NOT_STARTED (-2018L)
SL_RET_CODE_EVENT_LINK_NOT_FOUND (-2019L)

/* GENERAL ERRORS CODES*/
SL_ERROR_INVALID_OPCODE (-14337L)
SL_ERROR_INVALID_PARAM (-14338L)
SL_ERROR_STATUS_ERROR (-14341L)
SL_ERROR_NVMEM_ACCESS_FAILED (-14342L)
SL_ERROR_NOT_ALLOWED_NWP_LOCKED (-14343L) /* Device is locked, Return to Factory Image or
    Program new image should be invoked (see sl_FsCtl, sl_FsProgram) */


/* SECURITY ERRORS CODE */
SL_ERROR_LOADING_CERTIFICATE_STORE (-28673L)

/* Device is Locked! Return to Factory Image or Program new
image should be invoked (see sl_FsCtl, sl_FsProgram) */
SL_ERROR_DEVICE_LOCKED_SECURITY_ALERT (-28674L)

SL_ERROR_LENGTH_ERROR_PREFIX (-30734L)
SL_ERROR_WAKELOCK_ERROR_PREFIX (-30735L)
SL_ERROR_DRV_START_FAIL (-30736L)
SL_ERROR_VALIDATION_ERROR (-30737L)
SL_ERROR_SETUP_FAILURE (-30738L)
SL_ERROR_HTTP_SERVER_ENABLE_FAILED (-30739L)
SL_ERROR_DHCP_SERVER_ENABLE_FAILED (-30740L)
SL_ERROR_WPS_NO_PIN_OR_WRONG_PIN_LEN (-30741L)
```

32 Root certificate catalog

The following list of root CA can be verified using the on-board root certificate catalog.

ACEDICOM Root
Actalis Authentication Root CA
AddTrust Class 1 CA Root
AddTrust External CA Root
AddTrust Qualified CA Root
Amazon Root CA 1
Amazon Root CA 2
Amazon Root CA 3
Amazon Root CA 4
ANF Global Root CA
Apple Root CA - G2
Apple Root CA - G3
Apple Root CA
Apple Root Certificate Authority
ApplicationCA2 Root
Atos TrustedRoot 2011
Autoridad de Certificacion Firmaprofesional CIF A62634068
Baltimore CyberTrust Root
Buypass Class 3 Root CA
CA Disig Root R1
CA WoSign ECC Root
Certigna
Certinomis - Root CA
CFCA EV ROOT
Chambers of Commerce Root - 2008
China Internet Network Information Center EV Certificates Root
Cisco Root CA 2048
Class 2 Primary CA
COMODO Certification Authority
COMODO ECC Certification Authority
COMODO RSA Certification Authority
ComSign Global Root CA
ComSign Secured CA
Cybertrust Global Root
D-TRUST Root Class 3 CA 2 EV 2009
DigiCert Assured ID Root CA
DigiCert Assured ID Root G2
DigiCert Assured ID Root G3
DigiCert Global Root CA
DigiCert Global Root G2
DigiCert Global Root G3
DigiCert High Assurance EV Root CA
DigiCert Trusted Root G4
DST Root CA X3

EE Certification Centre Root CA
Entrust Root Certification Authority - EC1
Entrust Root Certification Authority - G2
Entrust Root Certification Authority
Equifax Secure Certificate Authority
GeoTrust Global CA
GeoTrust Primary Certification Authority - G2
GeoTrust Primary Certification Authority - G3
GeoTrust Primary Certification Authority
GeoTrust Universal CA 2
GeoTrust Universal CA
GlobalSign ECC Root CA - R4
GlobalSign ECC Root CA - R5
GlobalSign Root CA - R2
GlobalSign Root CA - R3
GlobalSign Root CA
Go Daddy Root Certificate Authority - G2
Hellenic Academic and Research Institutions RootCA 2011
Hongkong Post Root CA 1
IdenTrust Commercial Root CA 1
KISA RootCA 1
Microsec e-Szigno Root CA 2009
OISTE WISeKey Global Root GB CA
QuoVadis Root CA 2 G3
Root CA Generalitat Valenciana
S-TRUST Universal Root CA
SecureSign RootCA11
SecureTrust CA
Staat der Nederlanden EV Root CA
Staat der Nederlanden Root CA - G2
Staat der Nederlanden Root CA - G3 Starfield Class 2 Certification Authority
Starfield Root Certificate Authority - G2
Starfield Services Root Certificate Authority - G2
StartCom Certification Authority G2
StartCom Certification Authority
Swisscom Root CA 1
Swisscom Root CA 2
Swisscom Root EV CA 2
SwissSign Gold Root CA - G3
SwissSign Platinum Root CA - G3
SwissSign Silver Root CA - G3
SZAFIR ROOT CA
SZAFIR ROOT CA2
T-TeleSec GlobalRoot Class 3
TeliaSonera Root CA v1
thawte Primary Root CA - G2
thawte Primary Root CA - G3
thawte Primary Root CA

TWCA Global Root CA
UCA Global Root
UCA Root
VeriSign Class 1 Public Primary Certification Authority - G3
VeriSign Class 2 Public Primary Certification Authority - G3
VeriSign Class 3 Public Primary Certification Authority - G3
VeriSign Class 3 Public Primary Certification Authority - G4
VeriSign Class 3 Public Primary Certification Authority - G5
VeriSign Class 4 Public Primary Certification Authority - G3
VeriSign Universal Root Certification Authority
Visa Information Delivery Root CA

List of Figures

List of Tables

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

Max-Eyth-Straße 1
74638 Waldenburg
Germany

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