



CMi6110
User's Manual
English
V1.0

Contents

1	DOCUMENT NOTES	3
1.1	COPYRIGHT AND TRADEMARK	3
1.2	CONTACTS.....	3
2	IMPORTANT USAGE AND SAFETY INFORMATION	4
3	USING THIS MANUAL	5
3.1	PURPOSE AND AUDIENCE	5
3.2	ONLINE RESOURCES	5
3.3	SYMBOLS.....	5
4	INTRODUCTION.....	6
4.1	PURPOSE.....	6
4.2	APPLICATION DESCRIPTION.....	6
4.3	PRODUCT FEATURES.....	6
4.4	COMPATIBILITY.....	7
5	GETTING STARTED.....	8
5.1	PURPOSE.....	8
5.2	PRODUCT SPECIFICATION CMi6110	8
5.3	MOUNT AND START-UP THE DEVICE.....	9
5.3.1	<i>Mounting and connection</i>	<i>9</i>
5.3.2	<i>Start-up and LED indications.....</i>	<i>9</i>
5.3.3	<i>Switch off/reboot module.....</i>	<i>9</i>
6	INTEGRATION GUIDE	11
6.1	PURPOSE.....	11
6.2	INTRODUCTION.....	11
6.3	STATUS AND CONFIGURATION PARAMETERS.....	11
6.3.1	<i>L+G UH50 error codes</i>	<i>16</i>
7	ADMINISTRATION REFERENCE.....	18
7.1	PURPOSE.....	18
7.2	SECURITY AND ACCESS CONTROL.....	18
7.3	APP CONFIGURATION OPTIONS.....	18
7.4	METER DATA TRANSMISSIONS.....	19
7.4.1	<i>M-Bus-encoded telegram.....</i>	<i>21</i>
7.4.2	<i>JSON-encoded telegram.....</i>	<i>24</i>
7.4.3	<i>SenML/CBOR encoded telegram.....</i>	<i>25</i>
8	TECHNICAL SPECIFICATIONS.....	30
9	TYPE APPROVALS.....	32
10	DOCUMENT HISTORY	33
10.1	VERSIONS.....	33
11	REFERENCES.....	34
11.1	TERMS AND ABBREVIATIONS	34
11.2	NUMBER REPRESENTATION	34

1 Document notes

All information in this manual, including product data, diagrams, charts, etc. represents information on products at the time of publication, and is subject to change without prior notice due to product improvements or other reasons. It is recommended that customers contact Elvaco AB for the latest product information before purchasing a CMi Series product.

The documentation and product are provided on an “as is” basis only and may contain deficiencies or inadequacies. Elvaco AB takes no responsibility for damages, liabilities or other losses by using this product.

1.1 Copyright and trademark

© 2022 Elvaco AB. All rights reserved. No part of the contents of this manual may be transmitted or reproduced in any form by any means without the written permission of Elvaco AB. Printed in Sweden.

CMi Series is a trademark of Elvaco AB, Sweden.

1.2 Contacts

Elvaco AB
Kabelgatan 2T
434 37 Kungsbacka
SWEDEN
Phone: +46 300 30250
E-Mail: info@elvaco.com

Elvaco AB Technical Support
Phone: +46 300 434300
E-Mail: support@elvaco.se

Online: <http://www.elvaco.com>

2 Important usage and safety information

The following safety precautions must be observed during all phases of the operation, usage, service or repair of any CMi Series product. Users of the product are advised to convey the information to users and operating personnel and to incorporate these guidelines into all manuals supplied with the product. Failure to comply with these precautions violates safety standards of design, manufacture and intended use of the product. Elvaco AB assumes no liability for customer's failure to comply with these precautions.

CMi6110 receives and transmits radio frequency energy while switched on. Remember that interference can occur if the product is used close to TV sets, radios, computers or inadequately shielded equipment. Follow any special regulations and always switch off the product wherever forbidden, or when you suspect that it may cause interference or danger.

3 Using this manual

3.1 Purpose and audience

This manual provides all information needed to mount, connect, configure and integrate a CMi6110 NB-IoT module and targets system integrators.

This manual will provide device-specific information for CMi6110, such as status/configuration parameters and message formats, needed to integrate the module with a DM system and a receiving MD server.

It is meant to be used along with the common “Elvaco NB-IoT Module Integrators Manual”, which provides information about the bootstrapping process, device management, data transport and encryption.

3.2 Online resources

To download the latest version of this user's manual, or to find information in other languages, please visit <https://www.elvaco.com/>.

3.3 Symbols

The following symbols are used throughout the manual to emphasize important information and useful tips:





The Note symbol is used to mark information that is important to take into consideration for safety reasons or to assure correct operation of the meter connectivity module.



The Tip symbol is used to mark information intended to help you get the most out of your product. It can for example be used to highlight a possible customization option related to the current section.

The following symbols are used to provide information on how the product should be used:

Symbol	Description
	Waste electrical products should not be disposed of with household waste. Please recycle where facilities exist. Contact your Local Authority for recycling advice.
	Electrostatic-sensitive device. Please observe the necessary ESD protective measures when installing the MCM.

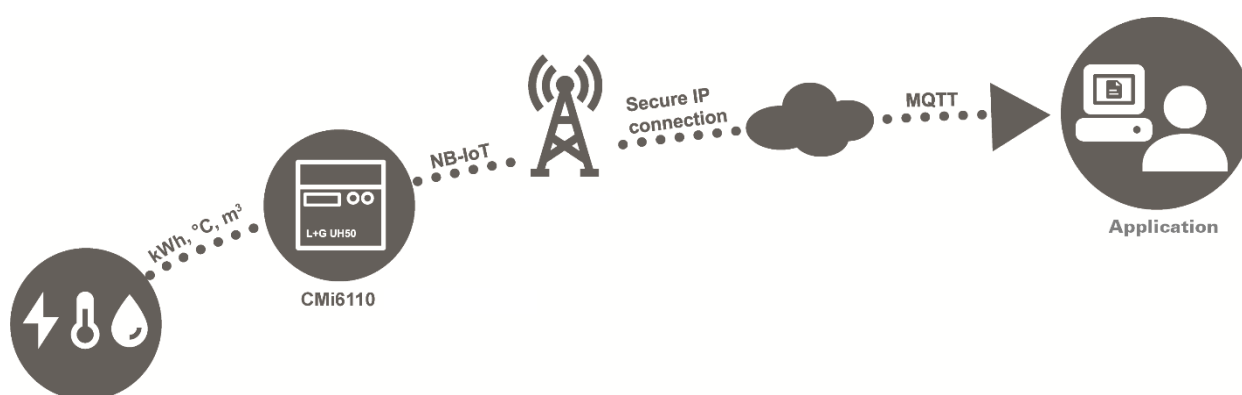
4 Introduction

4.1 Purpose

This chapter provides a general description of CMi6110. In the next-coming sections you will learn more about possible applications for the product and how it can be combined with other products to build versatile solutions.

4.2 Application description

CMi6110 is a cost-effective NB-IoT meter connectivity module, which is mounted inside a Landis+Gyr UH50 heat meter or UC50 calculator. As soon as the device has been mounted and deployed, it will start to deliver meter data to a receiving system via the NB-IoT (LPWAN) network. The product is ideal for applications where long range and high energy-efficiency are required and a lower bandwidth is not a concern.



4.3 Product features

Key features of CMi6110 include:

- IoT-ready**
 As soon as the meter connectivity module has been mounted and started up, it will automatically initiate transmission of meter data without any manual steps needed. The CMi6110 is prepared for seamless integration with all leading IoT platforms. It utilizes standards such as LWM2M, MQTT-SN and SenML-CBOR for fast and easy integration.
- Battery operated**
 CMi6110 has several options for power supply. It can be battery operated for up to 10 years with daily transmission of meter data.
- One-Touch Commissioning**
 The product uses the Elvaco One-Touch Commissioning (OTC) to configure and deploy products quickly and securely. Using the Elvaco OTC App, simply enter your desired settings and place your mobile phone on the right side of the UH50 meter/UC50 calculator. New settings will be applied instantaneously via NFC.
- Flexible message scheme**
 CMi6110 has different message formats to choose from, which makes it easy to setup the device for your specific project.

4.4 Compatibility

CMi6110 is compatible with any L+G UH50 meter using software version 5.15 or later and L+G UC50 calculators using software version 8.06 or higher.

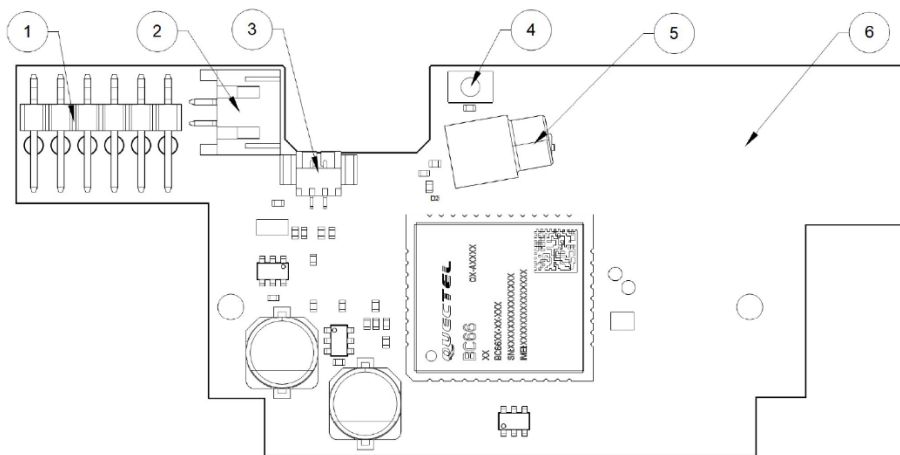
CMi6110 can be supplied by using one of the following PSUs: Elvaco's 230V CMip2110 or Landis+Gyr: WZU-110/AC230-xx, WZU-ACDC24-50 or battery WZU-NB-IoT-BAT.

5 Getting started

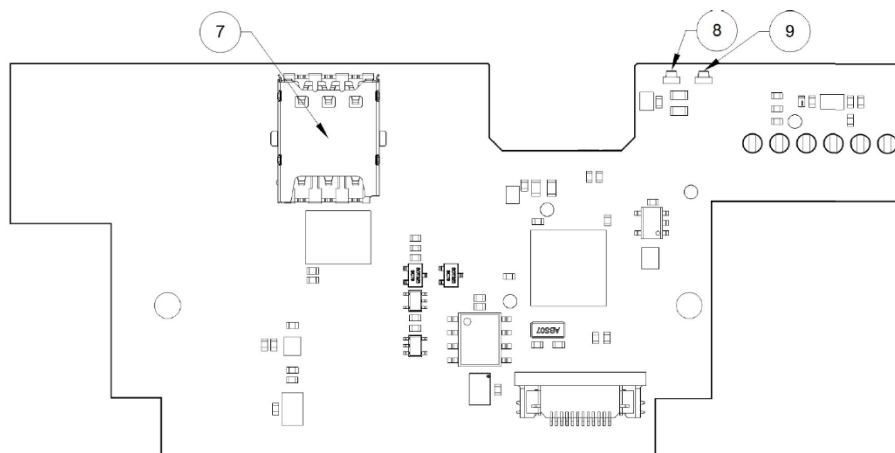
5.1 Purpose

This chapter provides instructions on how to get started with the CMi6110. After reading and carefully following each step of this chapter, the MCM will be mounted and deployed.

5.2 Product specification CMi6110



1. Meter Interface
2. PSU Power Connector
3. Battery Power Connector
4. Push Button
5. Antenna Connector (MCX female)
6. NFC Antenna



7. SIM card holder
8. Green LED
9. Red LED

5.3 Mount and start-up the device

5.3.1 Mounting and connection

In order to use CMi6110, a SIM card (size: Nano) needs to be mounted in the SIM card holder (8). The module is thereafter mounted in module slot 2 of a L+G UH50 heat meter or a L+G UC50 heat calculator with software version 5.15/8.06 or higher. Grab the module by the outer edges and gently press it into position. Make sure to connect the (longer) 2-wire cord from the meter power supply unit (110/230V) to the power connector and an external antenna to the module MCX connector.



Note that the battery supplying the meter must be connected before the battery supplying the NB-IoT module is connected to the CMi6110.

5.3.2 Start-up and LED indications

Module activation

Upon delivery, CMi6110 will be set to passive mode, which means that no messages will be transmitted from the module. Please make sure a SIM card (size: Nano) has been mounted before activating the module. There are two ways to activate the module:

1. Press down the push button for at least 5 seconds until the green LED lights up, then release the button. CMi6110 will confirm start-up by flashing its red and green LEDs for one second. At the order process a configuration can be added and the product will utilize the settings of this configuration when activated.
2. Via the Elvaco mobile application. Go to **Apply mode**, set the power mode to "active" and press **Apply settings**. Place the phone on the right side of the meter. The mobile phone should vibrate three times. This indicates that settings have successfully been applied.

Network Connection

When activated, CMi6110 will attempt to connect to the mobile network. The phase is indicated by the green and red LED lights up for 1 second, followed by short flashes on the green LED until the module has joined the mobile network. When CMi6110 succeeds in connecting to the mobile network, the green LED will lighten up for 8 seconds, as illustrated by Figure 1.

If the module fails to join the mobile network, it will perform retries until it succeeds. The time between each attempt will increase for every attempt until it is performed once every day. A new join attempt cycle can be manually started anytime by using the push button to reboot the module or by deactivating and activating the module using the Elvaco OTC App.

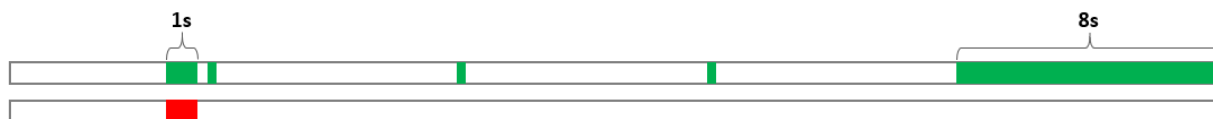


Figure 1: LED indications, network connection

5.3.3 Switch off/reboot module

To reboot the module, press and hold the push button for 5-15 seconds. Release the button when the green LED is lit.

To switch off the module, press and hold the push button for 15-20 seconds. Release the button when the red LED is lit.

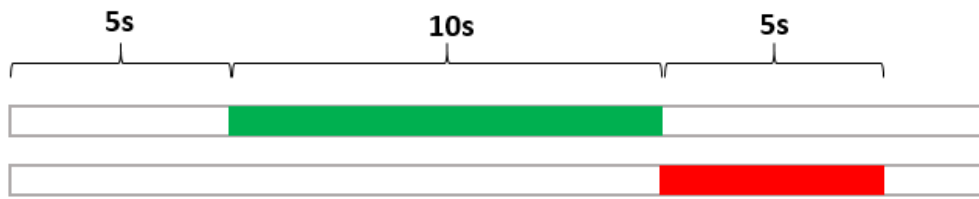


Figure 2: LED indication, reboot / switch-off

6 Integration guide

6.1 Purpose

This chapter provides the technical details needed to integrate an Elvaco NB-IoT module with a MD and/or DM server.



Note that this section will provide device-specific information and is meant to be used with the common “Elvaco NB-IoT MCM Integrator’s guide”.

6.2 Introduction

For device management, the module will act as a LWM2M client connecting to a Lwm2M server. The Device Management system enables configuration and monitoring of a CMi6110 module remotely. This includes setting configuration parameters, update the firmware and trigger momentaneous/historical readouts of the module. For meter data transport, the module uses the MQTT-SN protocol.

Upon activation, the device will attempt to connect to its configured bootstrap server via the mobile (NB-IoT) network. When successful, the module will receive connection credentials, i.e. IP addresses to the DM server and the meter data server.

The module will thereafter connect to the DM server and perform a DTLS handshake to generate the session key used to encrypt the data that is transmitted between DM server and module. Note that using DTLS is optional, and the product also support unencrypted communication.

The module will thereafter connect to the MQTT-SN gateway and perform a DTLS handshake to generate the sessions keys used to encrypt the session key used to encrypt the meter data transport.

Each module has a security chip where a device-unique set of keys are stored. These are provisioned to the module during production. The UDP transport of both DM and MDM can be secured using DTLS 1.2. Either the pre-provisioned keys can be used, or new keys can be provisioned during the bootstrap phase.

6.3 Status and configuration parameters

Table 1 below provides a list of all CMi6110 status and configuration parameters accessible on Lwm2M.

Op.	Lwm2M object	Lwm2M resource	ID	Type	Range or Enumeration	Comment
R	Lwm2M Security	LWM2M Server URI	0/0/0	String		Bootstrap URI
R	Lwm2M Security	Bootstrap server	0/0/1	Bool		TRUE
R	Lwm2M Security	Security Mode	0/0/2	Integer	0..4	BS Security mode 0 = PSK mode 3 = No security
R	Lwm2M Security	PSK Identity	0/0/3	Opaque		DevEUI
-	Lwm2M Security	Secret Key	0/0/4	Opaque		Bootstrap PSK
R	Lwm2M Security	Short Server ID	0/0/10	Integer	1..65534	
R	Lwm2M Server	Short Server ID	1/0/0	Integer	1..65534	
R	Lwm2M Server	Lifetime	1/0/1	Integer		
E	Lwm2M Server	Bootstrap-Request Trigger	1/0/9			
R(W)	Lwm2M Security	LWM2M Server URI	0/1/0	String		DM ServerURI Writable by Bootstrap server
R	Lwm2M Security	Bootstrap server	0/1/1	Bool		FALSE

Op.	LwM2M object	LwM2M resource	ID	Type	Range or Enumeration	Comment
R(W)	LwM2M Security	Security Mode	0/1/2	Int	0..4	DM Security mode Writable by Bootstrap server
R	LwM2M Security	PSK Identity	0/1/3	Opaque		DM PSK identity (DevEUI)
(W)	LwM2M Security	Secret Key	0/1/4	Opaque		DM PSK Writable by Bootstrap server
R	LwM2M Security	Short Server ID	0/1/10	Integer	1..65534	
R	LwM2M Server	Short Server ID	1/1/0	Integer	1..65534	
R	LwM2M Server	Lifetime	1/1/1	Integer		DM lifetime
E	LwM2M Server	Registration Update Trigger	1/1/8			
R	Device	Manufacturer	3/0/0	String		Manufacturer ("Elvaco")
R	Device	Model Number	3/0/1	String		Product model ("CMi6110")
R	Device	Serial Number	3/0/2	String		DevEUI
R	Device	Firmware Version	3/0/3	String		Firmware version
E	Device	Reboot	3/0/4			Reboot
R	Device	Available Power Sources	3/0/6/0	Integer	0..7	Power source 1: Internal battery 2: External battery 6: AC (Mains) power
R	Device	Power Source Voltage	3/0/7/0	Integer		Power source voltage (Millivolt)
R	Device	Battery level	3/0/9		0..100	Battery level (in %)
R	Device	Error Code	3/0/11/0	0..8		Error codes, according to LwM2M 1
RW	Device	Current Time	3/0/13	Time		Current time
RW	Device	UTC Offset	3/0/14	String		UTC Offset UTC+X (ISO 8601)
R	Device	Hardware version	3/0/18	String		Hardware version
R	Connectivity Monitoring	Network Bearer	4/0/0	Integer	0..50	7 = NB-IoT
R	Connectivity Monitoring	Available Network Bearer	4/0/1/0	Integer	0..50	7 = NB-IoT
R	Connectivity Monitoring	Radio Signal Strength	4/0/2	Integer		RSRP (NRSRP)
R	Connectivity Monitoring	APN	4/0/7/0	String		APN
R	Connectivity Monitoring	Cell ID	4/0/8	Integer		Cell ID
R	Connectivity Monitoring	SMNC	4/0/9	Integer	0..999	MNC PLMN = SMNC + SMCC

Op.	LwM2M object	LwM2M resource	ID	Type	Range or Enumeration	Comment
R	Connectivity Monitoring	SMCC	4/0/10	Integer	0..999	MCC PLMN = SMNC + SMCC
W	Firmware Update	Package URI	5/0/1			Firmware Update URI
E	Firmware Update	Update	5/0/2			Firmware Update Trigger
R	Firmware Update	State	5/0/3	Integer	0..3	Firmware Update Status 0: Idle 1: Downloading 2: Downloaded 3: Updating
R	Firmware Update	Update result	5/0/5	Integer		Firmware Update Result
R	Firmware Update	Firmware Update Protocol Support	5/0/8/0	Integer	0..5	0 = CoAP
R	Firmware Update	Firmware Update Delivery Method	5/0/9	Integer	0..2	0 = Pull only
R	LwM2M Cellular Connectivity	PSM Timer	10/0/4	Integer		NB-IoT T3412. Will be writeable in future releases.
R	LwM2M Cellular Connectivity	Active Timer	10/0/5	Integer		NB-IoT T3324. Will be writeable in future releases.
R	LwM2M Cellular Connectivity	eDRX parameters for NB-S1 mode	10/0/9	Opaque	8 bit	NB-IoT eDRX. Will be writeable in future releases. "This resource is encoded as octet 3 in [3GPP-TS_24.008, clause 10.5.5.32]."
R	LwM2M Cellular Connectivity	Activated Profile names	10/0/11	ObjLink		Link to APN Connection Profile object
RW	LwM2M APN Connection Profile	Profile name	11/[0,1]/0	String		
RW	LwM2M APN Connection Profile	APN	11/[0,1]/1	String		Manual APN Writable in object resource 1.
RW	LwM2M APN Connection Profile	Auto select APN by device	11/[0,1]/2	Boolean		Auto APN Mode Writable in object resource 1.
RW	LwM2M APN Connection Profile	Authentication Type	11/[0,1]/4	Integer	0..3	3 = None, Writing currently not supported
RW	Elvaco MDM Server	URI	33905/0/0	String		MDM Server URI

Op.	LwM2M object	LwM2M resource	ID	Type	Range or Enumeration	Comment
RW	Elvaco MDM Server	Protocol	33905/0/1	Integer	0..	MDM Server Protocol 0 = MQTT-SN
RW	Elvaco MDM Server	Transport Security Mode	33905/0/2	Integer	0..4	MDM Server Transport Security Mode 0 = PSK mode 3 = No security
W	Elvaco MDM Server	Transport Secret Key	33905/0/5	Opaque		MDM Server Transport Secret Key
RW	Elvaco MDM Server	Connection config	33905/0/10	Integer	0..1	MDM Server Connection Config 0: Optimized 1: Compliant
RW	Elvaco MDM Server	Topic	33905/0/11	String		MDM Server Topic
RW	Elvaco MCM Config	Meter readout interval	33906/0/0	Integer		Meter Readout Interval
RW	Elvaco MCM Config	Report data encoding	33906/0/1	Integer		Report Data Encoding 0: Reserved 1: JSON 2: MBus
RW	Elvaco MCM Config	Report frame type	33906/0/2	Integer		CMi6110: 42: Standard 43: Extended
RW	Elvaco MCM Config	Eco mode Enabled	33906/0/3	Boolean		
RW	Elvaco MCM Config	NFC Enabled	33906/0/4	Boolean		
R	Elvaco MCM Config	NFC Config-locked	33906/0/5	Boolean		
W	Elvaco MCM Config	Adjust time	33906/0/6	Integer		
E	Elvaco MCM Config	Instantaneous readout trigger	33906/0/10			Trigger a meter readout instantaneously.
E	Elvaco MCM Config	Historic resend trigger	33906/0/13			Execute arguments given as Unix timestamps: 0='<start-time>', 1='<stop-time>', 2='<min-interval>'
R	Elvaco MCM Config	Historic resend status	33906/0/14	Integer	0..50	Number of measurements queued for uplink
E	Elvaco MCM Config	Apply APN changes	33906/0/15			Stages changes in APN and APN mode, resets the device and tries to use those changes.
R	Elvaco NB-IoT Status	Uptime	33907/0/0	Integer		[s]
R	Elvaco NB-IoT Status	Average current consumption	33907/0/1	Integer		[uA]
R	Elvaco NB-IoT Status	Network classification	33907/0/2	Integer		0: Excellent 1: Good 2: Fair 3: Poor

Op.	LwM2M object	LwM2M resource	ID	Type	Range or Enumeration	Comment
R	Elvaco NB-IoT Status	ECL	33907/0/3	Integer	0..2	
R	Elvaco NB-IoT Status	RSSI	33907/0/4	Integer		[dBm *10]
R	Elvaco NB-IoT Status	SNR	33907/0/5	Integer		[dB*10]
R	Elvaco NB-IoT Status	MDM connection status	33907/0/10	Integer		0: OK 1: Connecting 2: No credentials 3: DTLS Rejected 4: MQTT-SN Failed 5: MQTT-SN Rejected
R	Elvaco Meter Info	Meter Model	33908/0/0	String		"UH50"
R	Elvaco Meter Info	Meter ID	33908/0/1	Integer		
R	Elvaco Meter Info	Comm status	33908/0/2	Integer		0: OK 1: No meter detected 2: Error
R	Elvaco Meter Info	Error flags	33908/0/3	Opaque		Meter Error flags
R	Elvaco NB-IoT Info	IMSI	33909/0/0	Integer		SIM IMSI
R	Elvaco NB-IoT Info	ICCID	33909/0/1	Integer		SIM ICCID
R	Elvaco NB-IoT Info	Registrations	33909/0/2	Integer		NB-IoT Registrations
R	Elvaco NB-IoT Info	Last Registration duration	33909/0/3	Integer		NB-IoT Last Registration Duration
R	Elvaco NB-IoT Info	Modem Model	33909/0/4	String		NB-IoT Modem Model
R	Elvaco NB-IoT Info	Modem Firmware	33909/0/5	String		NB-IoT Modem Firmware

Table 1: CMi6110 status/configuration parameters

6.3.1 L+G UH50 error codes

Make sure to double check error codes: identifier and explanation with latest meter specification.

Bit-No	Identifier	Explanation
0	F0	Error during flow metering (e.g. Air in measuring pipe)
1	F1	Interruption of flow temperature sensor
2	F2	Interruption of return temperature sensor
3	F3	Electronic for temperature evaluation defective
4	F4 ¹	Meter battery empty
5	F5	Short-circuit flow temperature sensor
6	F6	Short-circuit return temperature sensor
7	F7	Fault in the internal memory (CRC)
8	F8	Error F1, F2, F5 or F6 pending for longer than 8h.
9	F9	Error in the electronics
10	F0V	Prewarning for soiling of the measurement tube
11	F7V ²	Correctable error in the internal memory EEPROM 2

Table 2: UH50 error codes (TKB3448 V1.1 2008-10-06)

6.4 Changing APN via the DM system

Since changing APN is a potentially hazardous operation that may render the device disconnected from the mobile network, there is a rollback functionality in place when changing the APN.

To change APN, write the APN to the resource /10/1/1 and set APN mode to manual in /10/1/2. Once done, stage the changes by executing /33906/0/15. When executed, the device will reset and try to use the new APN. If the device manages a successful bootstrapping, the new APN will be saved as the default. If a successful bootstrapping has not happened for some time, the device will roll back to the old APN and reset again.

7 Administration reference

7.1 Purpose

This chapter contains detailed information about configuring options for CMi6110.

7.2 Security and access control

CMi6110 has a configuration lock feature, which prevents unauthorized access to the module. When configuration lock has been enabled, a Product Access Key will be needed to access the device via NFC. The Product Access Key is claimed by the end-user to his One-Touch Commissioning (OTC) account via the Elvaco OTC App or the OTC web interface.

7.3 App Configuration options

CMi6110 is configured via the Elvaco OTC App, using NFC to transfer settings to the module. The Elvaco OTC App is downloaded via Google Play. When the application has been installed, you can log in by using the user symbol in the top right corner. This will give you access to all your configuration profiles and enable you to configure any locked devices that have been claimed to your account.



Note that the Elvaco OTC App is only compatible with Android phones with Android 5.0 or later.

Table 3 provides a summary of all readable/writeable settings in CMi6110.

Parameter	Description	Configurable values	Default value	Device access – Locked device & correct PAK <u>or</u> open device	Device access – No PAK
Meter ID	Meter identification (secondary ID) for UH50/UC50.	N/A	N/A	Readable	Readable
Power mode	Activation status of the module.	Passive, Active	Passive	Readable / Writeable	Readable
Configuration Lock	Locks the module to prevent unauthorized access.	Open, Locked	Open	Readable / Writeable	Readable
APN mode	Sets how APN settings is implemented in the module.	Auto, Manual	Auto	Readable / Writeable	Readable
APN	APN of network provider.	N/A	N/A	Readable / Writeable	Readable
PLMN	PLMN of the network provider. (From SIM)	N/A	N/A	Readable	Readable

Message format	Sets the payload of the data message from the module.	Standard, Extended	Standard	Readable / Writeable	Readable
Message encoding	Sets the encoding of the payload.	M-Bus, JSON	M-Bus	Readable / Writeable	Readable
Transmit interval	Number of minutes between each data transmission.	5-1440	60	Readable / Writeable	Readable
Bootstrap IP	IP address of the bootstrap server the module will connect to upon activation.	N/A	84.19.147.226 (Elvaco Bootstrap server)	Readable / Writeable	Readable
Bootstrap port	Port of the bootstrap server the module will connect to upon activation.	N/A	5694	Readable / Writeable	Readable
Bootstrap security	Sets the way data sent from the module is encrypted.	DTLS / no security	DTLS	Readable / Writeable	Readable
MQTT-SN connection	Connection type used when publishing messages to the MQTT-SN broker.	Optimized, compliant	Optimized	Readable / Writeable	Readable
MQTT-SN topic	Topic used when publishing messages to the MQTT-SN broker.	N/A	-	Readable / Writeable	Readable
UTC offset	UTC offset of the meter (in minutes).	-720 - 720	0	Readable / Writeable	Readable

Table 3: CMi6110 settings

7.4 Meter data transmissions

CMi6110 sends meter data messages according to its transmit interval settings. Meter read out is always related to meter's clock at time 00:00:00. Transmission time is randomized between read outs.

The user can customize the data sent from the module by configuring **the encoding** and **the message format** of the telegram:

- There are three options for data encoding: M-Bus, JSON and SenML/CBOR.
- By selecting a message format, the user can configure the meter registers that will be included in the telegram. In message format *Standard*, all meter registers in Table 4 is included in the telegram. In message format *Extended*, all meter registers of Table 4 **and** Table 5 will be included in the message.

Field	Description
Energy	Energy consumption (kWh) Mapped to OBIS 6.8
Volume	Volume (m ³) Mapped to OBIS 6.26
Power	Power (kW) Mapped to OBIS 6.4
Flow	Flow (m ³ /h) Mapped to OBIS 6.27
Fw temp	Forward temperature (°C) Mapped to OBIS 6.29
Rt temp	Return temperature (°C) Mapped to OBIS 6.28
Meter ID	Identification number of the meter in which the module is installed
Error flags	Error and warning flags 16-bit hexadecimal value
Meter date/time	Date/time of the meter at time of readout

Table 4: Meter registers, standard message

Field	Description
Tariff 1	Energy consumption (kWh) Mapped to OBIS 6.8.1 Tariff register 1 or cooling energy (in combined heat/cooling meters)
Tariff 2	Energy consumption (kWh) Mapped to OBIS 6.8.2 Tariff register 2 or heating energy (in combined heat/cooling meters)
Tariff 3	Energy consumption (kWh) Mapped to OBIS 6.8.3 Tariff register 3
Missing time	Accumulated downtime.

Table 5: Meter registers, extended message

7.4.1 M-Bus-encoded telegram

In message format M-Bus, data will be M-Bus encoded. Data will be divided into Data Information Blocks (DIB) that include: Data information field (DIF code), Value information field (VIF code) and a data field (DATA) where the actual payload is stored (illustrated in Figure 3)



Figure 3: DIB structure

Table 6 provides a detailed description of how data is encoded when using this message format.

DIB	Field	Size	Data type	Description
1	Energy	6-7 bytes	INT32	Energy consumption (Wh, J) 0406xxxxxxxx = xxxxxxxx * 0.001 MWh (kWh) 0407xxxxxxxx = xxxxxxxx * 0.01 MWh 04FB00xxxxxxxx = xxxxxxxx * 0.1 MWh 04FB01xxxxxxxx = xxxxxxxx MWh 040Exxxxxxxxx = xxxxxxxx * 0.001 GJ (MJ) 040Fxxxxxxxx = xxxxxxxx * 0.01 GJ 04FB08xxxxxxxx = xxxxxxxx * 0.1 GJ 04FB09xxxxxxxx = xxxxxxxx GJ
2	Volume	6 bytes	INT32	Volume (m ³) 0413xxxxxxxx = xxxxxxxx * 0.001 m ³ 0414xxxxxxxx = xxxxxxxx * 0.01 m ³ 0415xxxxxxxx = xxxxxxxx * 0.1 m ³ 0416xxxxxxxx = xxxxxxxx m ³
3	Power	5 bytes	INT16	Power (W) 022Bxxxxxx = xxxxxx * 0.001 kW (W) 022Cxxxxxx = xxxxxx * 0.01 kW 022Dxxxxxx = xxxxxx * 0.1 kW 022Exxxxxx = xxxxxx kW
4	Flow	5 bytes	INT16	Flow (m ³ /h) 023Bxxxxxx = xxxxxx * 0.001 m ³ /h 023Cxxxxxx = xxxxxx * 0.01 m ³ /h 023Dxxxxxx = xxxxxx * 0.1 m ³ /h 023Exxxxxx = xxxxxx m ³ /h
5	Fw temp	4 bytes	INT16	Forward temperature (°C) 025Axxxx = xxxx * 0.1 °C 025Bxxxx = xxxx °C
6	Rt temp	4 bytes	INT16	Return temperature (°C) 025Exxxx = xxxx * 0.1 °C 025Fxxxx = xxxx °C

7	Meter ID	6 bytes	According to M-Bus EN13757-3 identification field	Meter ID 0C78xxxxxxxx
8	Error flags	5 bytes	INT16	Error and warning flags 02FD17xxxx For further information about Error flags please refer to table 2 or the latest meter's manual
9	Meter date/time	6 bytes	INT32	Meter date and time (YY-MM-DD HH:MM) Mapped to OBIS 9.36 046Dxxxxxxxx Bit 31-28 = Year-high* Bit 27-24 = Month Bit 23-21 = Year-low* Bit 20-16 = Day Bit 15 = Summer time flag** Bit 14-13 = Century Bit 12-8 = Hour Bit 7 = Error flag Bit 6 = Reserved for future use*** Bit 5-0 = Minute *The year is read by combining the year-high and year-low field. For example, year-high = 0010 and year-low = 010 => year = 0010010 **0 = standard time, 1= daylight-saving time ***0 = timestamp is valid, 1 = timestamp is not valid
10*	Tariff 1 Energy	7 bytes	INT32	Tariff 1 Energy consumption (Wh, J) 841003xxxxxxxx = xxxxxxxx Wh 841003xxxxxxxx = xxxxxxxx * 10 Wh 841003xxxxxxxx = xxxxxxxx * 100 Wh 841003xxxxxxxx = xxxxxxxx kWh 841003xxxxxxxx = xxxxxxxx *10 kWh 841003xxxxxxxx = xxxxxxxx MJ 841003xxxxxxxx = xxxxxxxx * 10 MJ
11*	Tariff 2 Energy	7 bytes	INT32	Tariff 2 Energy consumption (Wh, J) 842003xxxxxxxx = xxxxxxxx Wh 842003xxxxxxxx = xxxxxxxx * 10 Wh 842003xxxxxxxx = xxxxxxxx * 100 Wh 842003xxxxxxxx = xxxxxxxx kWh 842003xxxxxxxx = xxxxxxxx *10 kWh 842003xxxxxxxx = xxxxxxxx MJ 842003xxxxxxxx = xxxxxxxx * 10 MJ

12*	Tariff 3 Energy	7 bytes	INT32	Tariff 3 Energy consumption (Wh, J) 843003xxxxxxxx = xxxxxxxx Wh 843003xxxxxxxx = xxxxxxxx * 10 Wh 843003xxxxxxxx = xxxxxxxx * 100 Wh 843003xxxxxxxx = xxxxxxxx kWh 843003xxxxxxxx = xxxxxxxx *10 kWh 843003xxxxxxxx = xxxxxxxx MJ 843003xxxxxxxx = xxxxxxxx * 10 MJ
13*	Missing time	6 bytes	INT32	3C22xxxxxxxx = xxxxxxxx hours 3C23xxxxxxxx = xxxxxxxx days

Table 6: Payload, M-Bus encoded message

***Only included in the extended message.**

7.4.2 JSON-encoded telegram

The payload of message format JSON consists of one object with a list of key – value pairs. The names of each value type and unit is presented in Table 7. The values are encoded as numbers or strings and the units are encoded as strings.

Field	JSON key
Meter ID	ID
Meter date / time	TS
Energy	E
Energy unit	U
Volume	V
Volume unit	VU
Power	P
Power unit	PU
Flow	F
Flow unit	FU
Forward temperature	FT
Forward temperature unit	TU
Return temperature	RT
Return temperature unit	RU
Error flags	EF
Tariff 1 Energy*	T1
Tariff 1 Energy unit*	U1
Tariff 2 Energy*	T2
Tariff 2 Energy unit*	U2
Tariff 3 Energy*	T3
Tariff 3 Energy unit*	U2
Missing time*	MT
Missing time unit*	MU

Table 7: Payload, JSON encoded message

*Only included in the extended message.

Example payload, JSON:

```
{
  "TS": "2019-11-28T20:39Z",
  "ID": 87654321,
  "E": 12345.678,
  "U": "MWh",
  "V": 3456.7,
  "VU": "m3",
  "P": 5012,
  "PU": "W",
  "F": 212,
  "FU": "l/h",
  "FT": 80.3,
  "TU": "°C",
  "RT": 53.8,
  "RU": "°C",
  "EF": "0x4012"
}
```


7.4.3 SenML/CBOR encoded telegram

7.4.3.1 Introduction

For battery-powered devices it might be necessary to send several measurements in the same UDP frame to save energy. In order to achieve this SenML RFC 8428 - Sensor Measurement Lists (SenML) + CBOR RFC 8949: Concise Binary Object Representation (CBOR) is used to define a measurement list.

The idea is to send a list of measurements, where the first entry contains the base time for all the readouts (which only need to specify an offset) and the meter id shared by all readouts. The other records in the list may contain fewer readout fields to save space. The format allows sending all the data for every readout, in which case the save (in terms of bytes) is smaller and lies in that fewer telegrams are sent, some data needs not be transferred for every reading (like meter-id) and timestamps can be handled more efficiently. SenML/CBOR also provides one way to structure lists of readings in an efficient manner.

The first implementation will use M-Bus for encoding the data transferred, but other formats could be implemented in the future.

Note that SenML, CBOR and M-Bus are separate standards, this page describes how products can use these three in conjunction for representing multiple measurement values in a compact format suitable for radio transmission over for instance NB-IoT. Also, other means of encoding the data than M-Bus can be used in the future.

7.4.3.2 Elvaco Usage - Meter Readout Data Transfer

Elvaco uses SenML/CBOR/M-Bus data representation for transferring meter data in a compact and self-describing manner. The data being transferred is referred to as a pack, containing one record per readout.

Structure of SenML pack

Meter readout data is sent as SenML, i.e., a list (aka array) of readout values (records), encoded using CBOR. Each record is a map of key/value pairs using SenML.

Each product that uses the SenML/CBOR format shall follow the requirements below. In addition, it shall specify the exact contents of the data values included, meter id format etc. This specification alone is not sufficient for building a parser for a specific product.

Base Time

- *Base time* is used to set a reference time.
 - Timestamps are always encoded according to SenML (i.e., UNIX time). SenML label -1 "Base time", SenML definition of Time field
 - This value MUST be included in the first record of the pack
 - All other values have a *time* value that is added to the *base time* to define the exact time of the readout

Base Name

- *Base name* is used to represent the MeterID (Meter identification in M-Bus)
 - This value MUST be included in the first record of the pack
 - This is represented as a string array (CBOR Major Type 3 - SenML label -2 "Base name")
 - The product shall specify the exact format for this field, as it may vary depending on what type of "meter" is used. For an M-Bus format it is typically the M-Bus data without DIF/VIF.

- No *name* is set for remaining meter readout values, only values belonging to a single meter can be represented in one pack.

Data values

- The actual values from the meter can be encoded using multiple methods, such as M-Bus.
- The first record can also contain a data value field containing more information than the remaining records in the pack. This is to include more information for the first reading and then only a subset of values for the remaining records to save space. (SenML label 8 - "Data value")

Other values

- (*Base*) *Unit* is not used, since the unit is specified by the M-Bus data
- An "Encoder Version field" is used in a separate record to define the type and version of the encoded payload data.

Additional Records

All records in the SenML pack are expected to contain measurement values. If there is a need for transmitting additional information in the same pack additional records can be added. For such records the name field shall be used by defining a name of at least a single character. In SenML the *base name* and the *name* fields are appended to result in the final record name.

The *name* shall contain at least one one character outside [A-Fa-f0-9] which signifies non-hexadecimal representation, since meter-id is typically decimal/hexadecimal and this makes it easier to check the record name for validity.

If a parser finds a record with a *name* field like described above that it does not recognize it shall ignore the record.

The following additional records are currently used

Record	Name field	Comment
Encoder Type & Version	"V"	This field allows defining versions for the contents of the measurement field.

7.4.3.3 Encoder Type & Versioning

The following table defines allowed encoder types and versions. The information is sent in a special record "Encoder Version field".

- This field encapsulates both the encoding of the data and versioning
- It contains no timestamp
- It is encoded as a SenML Value
- It has a *Name* field with the single letter "V"
- If, when parsing, an invalid version is encountered the parsing shall stop with an error
- The value shall be interpreted as an UINT16
 - The first byte is the *encoder* type and the second is the *encoder version*, both interpreted as UINT8.
 - **Example:** value 0x0102 means Encoder type 0x01 and Encoder version 0x02.
 - Defined valid encoder types and versions are found in a table below on this page
 - Size of whole record is maximum 7 bytes

- If we ever need to extend this beyond 256 encoder types or versions, we could use an UINT32 and let the least significant byte overlap with the definition above and thus simply extend encoder type and version to use UINT16 instead of UINT8
- If record is excluded, *encoder* type is 0 and *encoder* version is 0

Record	Name field	Data	Comment
0 (M-Bus)	0	0x0000	M-Bus encoding of payload data. Each data record contains all DIF/VIF/Values according to M-Bus. Note that M-Bus uses LSB first byte order for the data and it shall be preserved here as well.

7.4.3.4 Example and Data Size

Below is a break-down of the number of bytes used for the different parts described above.

```

1                                     : size (bytes)
2 98 18                               # 24 item array                : 2 (fixed)
3  A3                                  # Map with length 3          : 1 (fixed)
4  21                                  # Key 1 = -2 = Base name     : 1 (fixed)
5  68                                  # Value 1 = String array, length 8 : 1 (fixed)
6      3132333435363738                # meter specific encoding    : 8 (fixed, depends on meter)
7  22                                  # Key 2 = -3 = Base time     : 1 (fixed)
8  1A 5DE02740                          # Value 2 = 1574971200 =    : 5 (fixed)
9                                     # Time "2019-11-28T20:00Z"
10  08                                  # Key 3 = 8 = Data value     : 1 (fixed)
11  58 21                               # Value 3 = Byte array, length 33 : 2 (payload1 < 256 bytes)
12                                     #                               or
13                                     #                               3 (payload1 > 255)
14      04064E61BC000415
15      07870000022B9413
16      023BD400025A2303
17      025E1A0202FD1712
18      40
19                                     : variable
20                                     Sum : 22 + (1) + payload1 bytes
21
22      ** Record for defining encoder and version **
23  A2                                  # Map with length 2          : 1 (fixed)
24  00                                  # Key 1 = "0" name           : 1 (fixed)
25  61 56                               # Value 1 = string => "v" = version : 2 (fixed)
26  02                                  # Key 2 = integer value      : 1 (fixed)
27  00                                  # Value 2 UINT16
28                                  # 0x0000 => enc=0, ver=0     : 3 (max)
29                                     Sum : 8 bytes (max)
30      ** Follows X items of same size **
31
32  A2                                  # Map with length 2          : 1 (fixed)
33  06                                  # Key 1 = 6 = Time           : 1 (fixed)
34  39 0E0F                             # Value 1 = -3600 =          : 3 (fixed)
35                                     # Time "2019-11-28T19:00Z"
36  08                                  # Key 2 = 8 = Data value     : 1 (fixed)
37  46                                  # Value 2 = Byte array, length 6 : 1 (payload < 24)
38      0406F24FBC00                    # M-bus record with one DIB: : variable
39                                     # Energy = 12341,234 MWh
40                                     Sum : X * (7 + (1) + payload2 size)
41
42      Total: 22 + (1) + payload 1 + 8 + X * (7 + (1) + payload2 size)

```

Given the fixed sizes above using M-Bus and assuming payload is < 256 bytes for the first record and < 24 for the subsequent records, the total size is:

$$29 + \text{payload1} + 6 + X * (7 + \text{payload2})$$

Some example sizes:

payload1	payload2	Total #records	Total size
33	6	24	367
33	33	12	508
36	32	24	968

7.4.3.5 Validators

<http://cbor.me/> - Validator for CBOR, does not understand SenML or M-Bus

Noted a small bug in the hex interpretation of negative numbers, the diagnostic window seems correct though.

7.4.3.6 Configuration

SenML/CBOR is to be considered a *message encoding*. It defines how the messages are encoded, but not the actual contents of the messages (which fields from the meter are included). SenML/CBOR/M-Bus is one such encoding, but there could be several based on this SenML/CBOR specification and the *encoder version field* above defines exactly which type and version is used.

The contents of the message are defined by the *message format*. The message format sets which fields are to be included in both the first and the subsequent records of the SenML pack.

The number of records included in a pack is set by the report and transmit intervals. See Scheduling Readouts for more details. If the readout interval is 120 minutes and the transmit interval is 1440 minutes 12 readouts in total will be included.

7.4.3.7 Message Size Restrictions

Each product may have different maximum payload sizes in a single telegram. Also depending on configuration (DTLS or not for instance) the net payload size may vary. Therefore, the device shall “fill up” as many telegrams as required to send the data. It is for the user to define a configuration that gives a reasonable tradeoff between power consumption (send fewer telegrams) and functional requirements (much data is sent).

If a device is configured using a *Message Format* and many readouts the data may not fit in a single telegram. In such cases multiple telegrams shall be sent and each telegram shall be fully self-described, i.e., contain Meter ID, timestamps etc.

Examples

Example 1:

Parameter	Value
Report interval	60
Transmit interval	1440 (daily)
Message encoding	SenML/CBOR/M-Bus version 0
Message format	Standard
Max transmissions per day	3

This example results in the transmission of one message per day, containing 24 readings, all with the contents defined in the Standard message format. Data is encoded using SenML/CBOR/M-Bus. Maximum 3 unsent such messages are sent each time (if for some reason the messages were not sent

"last time"). So maximum transmitted messages per day is 3 (containing $3 \times 24 = 72$ readings, covering 3 days)

Example 2:

Parameter	Value
Report interval	120
Transmit interval	720
Message encoding	SenML/CBOR/M-Bus version 0
Message format	Tariff
Max transmissions per day	2

This example results in the transmission of one message every 12h, containing 6 readings, all with the content defined in the Tariff message format. Data is encoded using SenML/CBOR/M-Bus. Maximum 2 unsent such messages are sent each time (if for some reason the messages were not sent "last time"), so maximum transmitted messages per day is 4 (containing $4 \times 6 = 24$ readings, covering 2 days).

8 Technical specifications

Type	Value	Unit	Comments
Mechanics			
Dimensions (w x h x d)	84 x 37 x 12	mm	
Weight	17	g	
Mounting	In Landis+Gyr UH50 / UC50 module slot 2	-	
External antenna connector	MCX female	-	
SIM card	Slide, size Nano	-	
Electrical connections			
Supply voltage	PSU & Battery Expected battery life time with ECO is 7+1 year with daily transmission of hourly values using SenML-CBOR	-	Elvaco PSU: CMip2110 Landis +Gyr Pack: WZU-NB-IoT-BAT Landis +Gyr PSU: WZU-AC230-xx WZU-ACDC24-00
Electrical characteristics			
Nominal voltage PSU	3.0 - 4,2	VDC	
Nominal voltage Battery	2.4 – 3.4	VDC	
Power consumption (max)	400	mA	
Power consumption (sleep mode)	6	µA	
Environmental specifications			
Operating temperature	+5 to +55	°C	
Operating humidity	0 - 93	% RH	No condensation
Operating altitude	2000	m	
Pollution degree	Degree 1	-	
Usage environment	Indoors	-	
Storage temperature	-20 to +60	°C	Storage temperature for battery pack is separated. See info on specific battery pack.
Mobile network			
Transmit power	23.0	dBm	
Receiver sensitivity	-135	dBm	
Certified for Bands	20,8,3	-	Hardware support for: B1/B2/B3/B4/B5/B8/B12/B13/B17/B18/ B19/20/B25/B26/B28/B66/B71/B85
3GPP	Release 14 (NB2)	-	
User interface			
Green LED	Start-up, Network connection	-	
Red LED	Start-up, Error	-	
Push button	Start-up, reboot	-	

Configuration	<ul style="list-style-type: none"> NFC via Elvaco OTC App via LwM2M (Elvaco Evo DM-system, or third-party DM-system) Preconfig on delivery 	-	
General			
Supported Protocols	LwM2M, MQTT-SN	-	both over UDP
Security	DTLS 1.2	-	
Data storage			
Meter data storage	8000 meter read out	-	Stores all supported meter data

9 Type approvals

CMi6110 is designed to comply with the directives and standard listed below.

Approval	Description
Safety testing incl. CB certification	IEC 62368-1:2014 EN 62368-1:2014+A11
EMC Testing	ETSI EN 301 489-52 V1.1.0 _Draft (2016) ETSI EN 301 489-1 V2.1.1 (2017-02) EN 55032:2015 + A11:2020
Radio Equipment Directive (RED)	RED Certificate

10 Document history

10.1 Versions

Version	Date	Description	Author
V0.1	2021-02	Evaluation samples	Hampus Morberg
V1.0	2022-02	Rev 1B, FW 2.0.0	Christopher Vonasek

11 References

11.1 Terms and abbreviations

Abbreviation	Description
CBOR	Concise Binary Object Representation
COSE	CBOR Object Signing and Encryption
DevEUI	Device Extended Unique Identifier
DM	Device Management
DNS	Domain Name Server
DTLS	Datagram Transport Layer Security
IP	Internet Protocol
LPWAN	Low Power Wide Area Network
LWM2M	Lightweight Machine to Machine
MCM	Meter Connectivity Module
MD	Meter Data
MQTT	MQ Telemetry Transport
MQTT-SN	MQTT for Sensor Networks
NB-IoT	Narrowband Internet of Things
OSCORE	Object Security Constrained RESTful Environments
OTC	One-Touch Commissioning
PAK	Product Access Key
PSK	Pre-Shared Key
PSM	Power Save Mode
PSU	Power Supply Unit
SenML	Sensor Measurement List
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Universal Resource Identifier

11.2 Number representation

- Decimal numbers are represented as normal number, i.e. 10 (ten).
- Hexadecimal numbers are represented with prefix 0x, i.e. 0x0A (ten)
- Binary numbers are represented with prefix 0b, i.e. 0b00001010 (ten)