



**WIRELESS COMMUNICATION SYSTEM
NB-IoT**

NB-R4-V

Revision 1.0

Contents

1	Introduction	1
1.1	NB-IoT mobile data services	1
1.2	Module usage	1
1.3	Module properties	2
2	Overview of technical parameters	3
3	Configuration of the NB-R4-V module	4
3.1	Setting parameters of the NB-R4-V module using a configuration cable	4
3.1.1	Listing of configuration parameters of the NB-R4-V module	4
3.1.2	Overview of configuration commands of the NB-R4-V module ("HELP")	6
3.1.3	„System commands” group for general diagnostics	8
3.1.4	„Configuration” group of commands for writing of configuration	8
3.1.5	„System commands” group for control of module basic functions	9
3.1.6	Commands for setting communication with consumption meters and sensors	10
3.1.7	Commands for setting M-Bus meters	14
3.1.8	Commands for setting IEC 62056 ("OPTO") meters	17
3.1.9	Commands for setting meters with Modbus protocol	19
3.1.10	Commands for setting communication with the NB-IoT network	22
3.1.11	Commands of the "Utils" group for setting and checking basic module functions	25
3.2	Setting module parameters using an optical converter	29
3.3	Setting module parameters from a remote computer using the reverse channel	31
3.4	Data messages of the NB-R4-V module	32
3.4.1	Structure and types of module data messages	32
3.4.2	Description of INFO type message	33
3.4.3	Description of TRAP type message	34
3.4.4	Principle of message encryption	35
4	Operational conditions	36
4.1	General operational risks	36
4.1.1	Risk of mechanical and/or electric damage	36
4.1.2	Risk of premature battery discharge	36
4.1.3	Risk of damage by excessive humidity	36
4.2	The condition of modules on delivery	37
4.3	Modules storage	37
4.4	Safety precautions	37
4.5	Environmental protection and recycling	37
4.6	Module installation	37
4.7	Replacement of the module and replacement of the read meter	41
4.8	Dismantling the module	42
4.9	Module functionality check	42
4.10	Operation of the NB-R4-V module	42
5	Troubleshooting	43
5.1	Possible causes of system failures	43
5.1.1	Power supply failures	43
5.1.2	System failures	43
5.1.3	NB-IoT network communication failures	43
5.1.4	Communication faults with meters and sensors	44
5.2	Procedure for determining the cause of failure	45
6	Additional information	45

List of Tables

1	Overview of technical parameters of the NB-R4-V module	3
---	--	---

List of Figures

1	Appearance of the NB-R4-V module	2
2	Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (1)	30
3	Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (2)	30
4	Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (3)	31
5	Preview of „NEP coding table" for coding of variables in WACO system	32
6	Assembly of the NB-R4-V module with a rod antenna	38
7	Detail of the printed circuit board of the NB-R4-V module	38
8	Connection of a meter with RS-485 output to the NB-R4-V module terminal block	39
9	Diagram of connecting RS-485 bus wires to the NB-R4-V module terminal block	40

1 Introduction

This document describes the configuration options of the NB-R4-V radio module, which is used for reading the status of consumption meters and sensors with a physical output interface of type RS-485 and for radio transmission of information about the current readings of consumption meters and sensors to a remote reading system (Automatic Meter Reading - AMR) using NB-IoT services of a GSM operator.

1.1 NB-IoT mobile data services

Mobile data services NB-IoT are global data services provided by some operators of GSM services. The services are focused on the communication with a huge quantity of devices, that transfer only an extremely limited volume of data. Networks with such purpose and features are commonly labeled as „Internet of Things”, or by its acronym ”IoT”. NB-IoT (”Narrow Band Internet of Things”) is an open standard developed by 3GPP organization (3rd Generation Partnership Project) which is concerned with standardization in the GSM network development. NB-IoT is a cellular technology based on the LTE, that was developed specially for wireless communication with terminals of IoT category, that produces only limited volume of data, but they are miniature, inexpensive, with a very low energy consumption and they are commonly installed in the places with high demands on the signal coverage. Typical example of such device is a reading module of the water/gas/electro-meter installed in the basement without electricity, that should be able to run reliably many years on the internal battery even in weak signal conditions, where other services fail.

NB-IoT technology maximally utilizes technological infrastructure of LTE data services in licensed radio band. Combination of narrow frequency band and the most advanced modulation techniques enable increasing of receiver sensibility to the -135 dBm level, so that an existing infrastructure of mobile operator provides global coverage with high signal penetration even in build-up urban areas. Thus, the service is available in the places, where IoT category devices are typically installed - in shafts, distribution boards and cellars.

Terminal devices are identified in the network by standard SIM of GSM operator. Global system of SIM evidence and single communication standard enable providing of international services (roaming). Bi-directional communication is carried by standard Internet protocol with UDP transport layer. Messages are transferred from the GSM operator network to the IoT-terminal operator through the designated data gateway (Access Point - AP) either to public Internet, or to operator’s private IP network (i.e. same way as any similar mobile data services). Addressing and routing details depend on the network configuration and policy of particular GSM operator. Typical example of addressing and routing is a solution, when the GSM network automatically assigns private IP addresses to IoT terminals, IP-packets with messages are routed through the private IP network to a single Access Point, where they are re-addressed and resend through a single pre-arranged public IP-address to the public Internet. The IoT terminal assigns packets by target server public IP-address, that is preset in its configuration. Target system can identify original source of the message by using of device unique identifier (IMEI), which is a requisite part of the message content.

1.2 Module usage

The NB-R4-V module is designed for remote reading of electronic consumption meters (electricity meters, gas meters, heat meters) and sensors that are equipped with a data output for connection to an RS-485 bus with data encoding according to M-Bus, Modbus, or IEC 62056 standards. The module has one bus input of RS-485 type, to which up to six consumption meters or sensors of different types can be connected. The module regularly queries the current data values from connected devices via the RS-485 bus interface and sends this data to the superior remote reading system (AMR) in the form of radio messages of the NB-IoT service of a mobile operator (hereinafter referred to as ”INFO message”).

The NB-R4-V module can be used for reading **up to 6 meters located on one RS-485 data bus**, with the ability to read up to four selected variables (registers) from each connected meter with M-Bus, Modbus, or IEC 62056 encoding system. For each meter, the encoding system (protocol) and serial communication parameters can be individually set to suit the given meter type. The reading period can also be set separately for each meter. The module performs reading of the register contents of the given meter with the set period and either sends a radio message with the read values immediately (”online” mode), or stores the read values in memory for later bulk sending (”history” mode). The module has allocated memory for storing up to 100 variables that can be sent in one summary message. This way, for example, messages from four meters for six past reading periods can be sent in bulk (4 meters * 4 variables * 6 measurement intervals = 96). This method of communication is optimal both from the perspective of minimizing electrical energy consumption (the module is powered by a built-in battery) and from the perspective of minimizing the costs of NB-IoT services.

The module has a configuration table for introducing up to six meters, which are distinguished by bus identifiers according to the relevant standard. When querying the status of registers, the module uses these identifiers. If the connected meter with M-Bus or IEC 62056 encoding does not support bus addressing, the module can only query using a multicast address and cannot distinguish which meter the response came from. In this case, its use is limited to connecting only one meter. The module allows data transmission **in both open and encrypted mode**.

Messages are transmitted to the module operator's application server via the NB-IoT service in the form of standard IP packets routed to the user's IP network through an access point (Access Point) contractually defined between the GSM network operator and the module operator. The operator's application server decodes the messages and further processes the information contained in them.

The NB-R4-V module is equipped for **bidirectional communication** and is capable of receiving messages with commands in NEP format from a remote server via the GSM network. These messages can be used to set module parameters remotely, from a remote server.

1.3 Module properties

The NB-R4-V module is enclosed in a moisture-resistant plastic box (IP65 protection) and is suitable for use in both indoor and outdoor environments. The box is designed for wall mounting or mounting on any structural element (beam, pipe...). The module can be equipped with additional moisture protection (to IP68 degree) by filling with high-adhesion silicone. If this modification is required from the manufacturer, it must be ordered with a special order code.

The module is powered by an internal battery with a very long lifetime. When reading data from one meter with a period of 15 minutes and a data sending interval with a period of 6 hours, the battery life is longer than 7 years. The battery life can be negatively affected by frequent reading and sending of messages (for example, when operating in "online" mode), sending long INFO messages (for example, when sending data from multiple meters), as well as operating the device in facilities with temperatures outside the recommended operating temperature range.

The module is equipped with a SIM card holder for use with a "Micro-SIM" (3FF) format SIM card with dimensions of 15 x 12 x 0.76 mm. The SIM holder is located inside the module on the main board. The module can be manufactured on order with an integrated SIM module (chip-SIM) of a specific GSM operator.

The module can be controlled and configured using a configuration cable, or wirelessly, using an optical converter. For optical configuration, the module is equipped with a circular "peephole" to support magnetic attachment of the optical converter. The module can also be configured remotely, using the reverse channel of bidirectional communication.

The appearance of the NB-R4-V module is shown in Figure 1.



Figure 1: Appearance of the NB-R4-V module

2 Overview of technical parameters

An overview of the technical parameters of the NB-R4-V module is provided in Table 1.

Table 1: Overview of technical parameters of the NB-R4-V module

NB-IoT transmitter parameters		
Frequency band 800 MHz (RX/TX)	791-821 / 832-862	MHz
Frequency band 850 MHz (RX/TX)	869-894 / 824-849	MHz
Frequency band 900 MHz (RX/TX)	925-960 / 880-915	MHz
Modulation type	GMSK, 8PSK	(adaptive)
Bandwidth	180	KHz
Transmission power	200	mW
Receiver sensitivity	135	dBm
Communication protocol	NB-IoT	(bidirectional)
Transmission speed	0.35 ÷ 240	Kb/s (adaptive)
Antenna input characteristic impedance	50	Ω
Antenna connector	SMA female	
Data interface		
Bus interface	RS-485	(terminals "A/+", "B/-")
Transmission speed	300 ÷ 19200	Baud
Operation mode	asynchronous	
Transmission parameters	8 data bits, 1 stop bit, no parity	
Signal level	according to CCITT V.11	
Auxiliary terminal "GRD"	"ground"	"ground" connection
Supported data protocols	M-Bus, IEC 62056, Modbus	
Number of connected meters/sensors	6	
RS-232 configuration interface		
Transmission speed	9600	Baud
Operation mode	asynchronous	
Transmission parameters	8 data bits, 1 stop bit, no parity	
Signal level	TTL/CMOS	
Optical configuration interface		
Transmission speed	115 200	Baud
Optical wavelength	870	nm
Optical interface specification	complies with IrPHY 1.4 standard	
Power supply parameters		
Lithium battery voltage	3.6	V
Lithium battery capacity	17	Ah
Mechanical parameters		
Length (without antennas)	200	mm
Width	70	mm
Height	60	mm
Weight	approx. 250	g
SIM card dimensions	(15x12x0.76)mm	"Micro-SIM"
Storage and installation conditions		
Installation environment (according to ČSN 33 2000-3)	normal AA6, AB4, A4	
Operating temperature range	(-20 ÷ 40)	°C
Storage temperature range	(0 ÷ 40)	°C
Relative humidity *	95	% (without condensation)
Protection rating *	IP65 or IP68	

* modules with additional silicon filling sealing are waterproof, with IP68 degree of protection.

3 Configuration of the NB-R4-V module

Configuration parameters of the NB-R4-V module can be displayed and changed from the common computer (PC) or smartphone by one of these methods:

- with using of „**USB-CMOS**” converter and configuration cable connected to the module;
- wirelessly, with using of „**USB-IRDA**” or „**BT-IRDA**” converter;
- **remotely**, by using of bi-directional communication system.

Technique of interconnection of the module with configuration computer and general rules of configuration are described in detail in the chapter 2 of „**Configuration of wacoSystem product family devices**”, that can be downloaded from the producer website:

www.softlink.cz/en/documents/

The description and meaning of all configuration parameters that can be checked and changed by cable can be found in the section 3.1 „Setting of NB-R4-V parameters via configuration cable”.

Description of interconnection of the converter with PC („USB-IRDA”) or smartphone („BT-IRDA”) and general rules of configuration with using of **optical converters** are described in the chapter 3 of above mentioned manual „Configuration of wacoSystem product family devices”. The description and meaning of the parameters that can be changed by optical converter can be found in the section 3.2 „Setting of parameters by using of optical „IRDA” converter”.

Principles and short description of communication through the **NB-IoT reverse channel** can be found in paragraph 3.3 „Remote setting of module parameters through the NB IoT reverse channel”.

3.1 Setting parameters of the NB-R4-V module using a configuration cable

The following part of the manual describes those parameters of the NB-R4-V module whose current values can be checked with using of directly connecting the module to a PC using a configuration cable, and possibly changed by entering of the configuration commands (configuration ”from the command line”).

3.1.1 Listing of configuration parameters of the NB-R4-V module

The configuration parameters can be displayed by entering the command ”**show**” into the command line and pressing the ”ENTER” key.

The following parameter listing will appear in the terminal window:

```
cfg#show
----- Configuration -----
Timezone : 1
Server IP : '192.168.203.25'
Ping IP : '192.168.203.25'
Server port : 2001
Reply to server : no
My src port : 2000
APN : 'nb.telemetry.vf'
Band : 20

Ping IP : '192.168.203.25'
Max session time 172800 sec - 2d, 0:00:00
Error restart time : 24 hours
Main Send periode : 1440

Data will be unencrypted
```

```
----- Configuration 0 -----
MODBUS mode
Uart speed 9600 8E1
Meter address : 0
MBUS address : 00000000
MBUS version : 31
MBUS medium : 2
MBUS manufacturer : SFT
-- Register 1 --
  Reg address : 450, type INT48, func 3
-- Register 2 --
  Reg address : 514, type NONE, func 3
-- Register 3 --
  Reg address : 578, type NONE, func 3
-- Register 4 --
  Reg address : 642, type NONE, func 3
RT : 4 * 50ms
FT : 1 * 50ms
Resp : 20 * 50ms
iDel : 1 * 50ms
Repeat : 2
Send periode : 60 min.
Next send : 46 min.
```

```
----- Configuration 1 -----
MBUS mode
Uart speed 2400 8E1
Meter address : broadcast
MBUS max more packets : 0
  DIF/VIF[0] : 00 00
  DIF/VIF[1] : 00 00
  DIF/VIF[2] : 00 00
  DIF/VIF[3] : 00 00
RT : 30 * 50ms
FT : 1 * 50ms
Resp : 20 * 50ms
iDel : 10 * 50ms
Repeat : 2
Send periode : 0 min.
```

```
----- Configuration 2 -----
OPTO mode
Uart init speed 2400 7E1
Max speed : 19200
Meter address :
-- Register 1 --
  Reg value : C.1.0
-- Register 2 --
  Reg value : 1.8.1
-- Register 3 --
  Reg value : 1.8.2
-- Register 4 --
  Reg value : 2.8.0
RT : 4 * 50ms
FT : 1 * 50ms
Resp : 100 * 50ms
iDel : 3 * 50ms
Repeat : 2
Send periode : 0 min.
```

```

---- Configuration 3 -----
---- Configuration 4 -----
---- Configuration 5 -----

-- Narrow band modem --
Next send : 1426 min.
No. sent  : 0 msg(s)
No. rcv   : 0 msg(s)

Modem state : 2 - sleep
Session count : 0
Session timeout : 0 sec - 0:00:00
Restart timeout : 0 sec
Life timeout   : 0 sec
Modem IMEI  : 864898061519593
SIM CCID   : 89882390000727626486
SIM IMSI   : 901288910195883
Last RSSI  : 0 dBm
Last IP    : 0.0.0.0

Conf. version : 0
SW version 2.02, date Jun 19 2023
cfg#

```

In the introductory section of the configuration parameter listing, information about the module's communication system settings and some other setting parameters are displayed, the meaning of which is described below.

In the middle section of the listing, configurations of individual inputs are displayed. The module allows connection of up to six meters or sensors, so the configuration listing contains 6 sections ("Configuration 0" to "Configuration 5"). Each section contains settings for communication via the RS-485 interface (communication protocol, initial and maximum data rate, meter identification) and settings for reading individual variables.

In the last section of the module's configuration parameter listing, some **identification and operational data of the module** are displayed. At the beginning of this section are statistical data on message transmission via the NB-IoT network. The "**Next send**" item is the time until the next regular message is sent. The "**No. sent**" and "**No. rcv**" items contain statistics of received/sent messages.

This is followed by information about the NB-IoT subsystem settings - current status of the GSM modem "**Modem state**", GSM modem identification data "**IMEI**", inserted SIM card number "**SIM CCID**" and unique SIM card user number "**SIM IMSI**". The "**Last RSSI**" line shows the signal strength with which the last message from the GSM network was received, the "**Last IP**" line shows the last assigned IP address from the NB-IoT network. It also displays the number of established connections since the last reset "**Session count**", time until the maximum connection time expires "**Session timeout**", and time until GSM modem restart in case of connection failure "**Restart timeout**".

At the end of the listing, the configuration parameter set number "**Conf. version**" is displayed, which increases with each new configuration saving to memory (it is reset by erasing FLASH memory) and the software version "**SW version**" with its release date.

The procedure for setting individual parameters and a more detailed explanation of their meaning can be found below.

3.1.2 Overview of configuration commands of the NB-R4-V module ("HELP")

The overview of configuration commands ("HELP") and their parameters can be displayed by entering the command "?" into the command line and pressing the "ENTER" key. A "Help" listing will appear in the terminal window, containing a summary of all configuration commands divided into several sections - see the example below.

The meaning and usage of individual commands is explained in the following parts of section 3.1.

```

--- System commands ---
deb          : Show or set debug level
ta          : Show tasks
mb          : Show mail boxes
du addr     : Dump memmory
rb addr     : Read byte from addr
rw addr     : Read word from addr
rd addr     : Read dword from addr
sb addr val : Set byte on addr
sw addr val : Set word on addr
sd addr val : Set dword on addr
port        : Show port [a,b,..]
--- Configuration ---
show        : Show info
info        : Show module info
write       : Write configuration to flash
cread       : Read configuration from flash
clear       : Clear configation and load defaults
--- All profiles [0 - 5] ---
proto       : Set protocol per meter [0 - 5] 0 - opto, 1 - mbus, 2 - modbus
ispeed      : Communication speed
parity      : Parity N,0,E (bits)
periode     : Send periode in minute, 0 - disable
irt         : rising time * 50ms
ift         : falling time * 50ms
iresp       : responce time * 50ms
idel        : delay time * 50ms
irep        : Repeat readout
iread       : Readout BUS device
--- Opto protocol commands per meter [0 - 5] ---
oid         : Meter ID (0 - 99999999)
mspeed      : Communication max. speed
reg1        : Register for value 1
reg2        : Register for value 2
reg3        : Register for value 3
reg4        : Register for value 4
--- Wired MBUS commands per meter [0 - 5] ---
id          : MBUS address (0 - 255)
sid         : MBUS secondary address (0 - 99999999)
mcount      : max more packets (0 - 10)
var1        : DIF VIF variable
var2        : DIF VIF variable
var3        : DIF VIF variable
var4        : DIF VIF variable
--- ModBus protocol commands per meter [0 - 5] ---
id          : Meter address (0 - 255)
reg0        : Register for MBUS address
type0       : Value type for MBUS address (default 0 - NONE)
func0       : Readout command (1 - 4)
type1       : Value type (? for help)
func1       : Readout command (1 - 4)
reg1        : Register address for value 1
type2       : Value type
func2       : Readout command (1 - 4)
reg2        : Register address for value 2
type3       : Value type
func3       : Readout command (1 - 4)
reg3        : Register for value 1
type4       : Value type
func4       : Readout command (1 - 4)
reg4        : Register for value 1
sc          : Send modbus command

```

```

--- Narrow band ---
server      : Server IP address
sport      : Server UDP port
testip     : Ping IP address
sreply     : Send reply to server
apn        : Access Point Name
sess       : Set max session time in minutes
errtime    : Set restart time on error in hours
band       : Set NB band, default 20 - Europe
tshort     : Set modem short timeout
tlong      : Set modem long timeout
tconn      : Set modem connection timeout
sping      : Send ping
at         : modem command

--- Utils ---
tz         : Time offset in hours
ppm        : Set RTC ppm
xmco       : Enable/disable Xtal on MCO
xtset      : Set Xtal freq for ppm
hsical     : Calibrate HSI
time       : Show or set rtc time, set as BCD : 0x102033 is 10:20:33
date       : Show or set rtc date, set as BCD : 0x171231 is 2017-12-31
vbat       : Show or set vbat for alarm (vbat min)
uptime     : Show uptime
sens       : Show sensors
sendp      : Send x NB messages
send       : Send NB message
periode    : Send periode 0 - disable, >0 periode in minutes
hist       : History periode 0 - disable, >0 periode in minutes
ekey       : Set encrypt key NEP, point '.' no encrypt
mint       : ADC measure interval in sec.
loca       : Show or set location (0-30 chars)
reset      : Reset device
hdata      : show data for send
?          : Show this help
cfg#

```

3.1.3 „System commands” group for general diagnostics

Commands „deb”, „ta”, „mb”, „du addr”, „rw addr”, „rb addr”, „rd addr”, „sw addr val”, „sb addr val”, „tshort”, „tlong”, „port”, „ppm” and „at” are used for troubleshooting and repair of the device in a factory. **Manufacturer strongly recommends not to use these commands during common operation.**

3.1.4 „Configuration” group of commands for writing of configuration

The module contains two sets of configuration: operating configuration and saved configuration. At the start of the system the module copies saved configuration to operating configuration, with which continues to work. If the user changes configuration parameters, it does so only in operating configuration.

If the current operating configuration was not stored to FLASH memory, the module returns to the saved configuration after reset. If the parameter should be changed only temporarily (for example shorten of the broadcasting period during installation), it is not necessary to save operating configuration into FLASH memory (after finishing a work the module can be returned to normal configuration by its reset). If the parameter should be changed permanently, there is necessary to save configuration to FLASH memory.

If operating configuration corresponds to the saved set (ie. there are no differences between commands in FLASH and in the operating set), the module will „report“ prompt in the format „cfg#”. If operating configuration was changed so that it no longer matches to the saved set, the module will report prompt in the format „cfg#”.

Every time the current configuration is saved into FLASH memory the value of the „Configuration version” parameter increases by one and the prompt changes to „cfg#”. The parameter resets to zero by erasing of FLASH.

Current operating configuration can be displayed by using of **"show"** command (see paragraph 3.1.1):

```
cfg#show
```

Current operating configuration can be rewrite the to FLASH memory by using of **"write"** command:

```
cfg#write
Writing config ... OK, version 13
cfg#
```

Reading of the configuration from FLASH memory can be done by using of **"cread"** command:

```
cfg#cread
Reading config ... OK, version 13
cfg#
```

The configuration can be erased in Flash memory by using of **"clear"** command:

```
cfg#clear
Clearing config ... OK, version 13
cfg#
```

This command deletes all configuration parameters from the FLASH memory, so it is necessary to set them again. If after erasing all parameters in FLASH memory the module goes to reset, default set of parameters (configured in the program of the device) is duplicated to FLASH memory. There is only one exception - frequency constant keeps the actual value also after cleaning of FLASH memory by "clean" command.

This command is recommended to use only by users with good knowledge of the system or after consultation with the manufacturer.

3.1.5 „System commands” group for control of module basic functions

This group of commands enables control of basic functions of the module. There are following commands:

?	<i>show list of configuration commands („Help”)</i>
reset	<i>command for module reset</i>
send	<i>immediate sending of radio message</i>
sendp	<i>immediate sending of series of messages</i>
sens	<i>show current values of internal sensors (temperature, voltage..)</i>
uptime	<i>show system uptime from last reset</i>
info	<i>show module info</i>

By **"?"** command the list of all configuration commands with their brief description ("Help") can be displayed. Example of using this command can be found in the initial part of section 3.1.

The command **"reset"** performs the module reset. After each reset the system starts with the parameters that are stored in FLASH memory. If the current configuration should be used after reset, it is necessary to store it into the FLASH before reset (see paragraph 3.1.4). Example of using of "reset" command:

```
cfg#reset
-- Reset code 0x14050302 --
PIN Reset
SFT Reset
SW version 0.01, date Jan 18 2019
Monitor started ..
cfg#
```

The command **"send"** can be used for immediate („out of turn”) transmitting of the standard information message with measured values. This command can be used for checking of radio signal availability during the system installation, or for any adjustments and testing of the module. The command makes possible to send the information message anytime without necessity to change the transmission period or without waiting until the message will be sent spontaneously within the pre-set period. Example:


```
cfg#send
Sending ...
send [1] msg 255
cfg#
```

The command **"sendp"** can be used for immediate transmitting of series of standard messages with 1-minute interval. This command can be used for checking of radio signal availability during the system installation. It could enable checking of connection also after closing of mounting rack, or after leaving of watemeter shaft. Number of transmitted messages is set by parameter (number) after command, the first message is transmitted immediately after command. Example of sending of series of 5 messages:

```
cfg#sendp 5
sending 5 msgs
cfg#
```

The **"sens"** command can be used for displaying of current values of A/D converters measuring physical quantities (battery voltage, temperature...). This command is intended only for module checking and diagnostics.

```
cfg#sens
-- Sensors --
CPU : 25.8 °C
VDA : 3.003 V
VBAT : 3.561 V
Sensor type 0
cfg#
```

The **"uptime"** parameter value shows the time interval passed from the last device reset in seconds so that the exact moment of the last module reset can be recognized by this parameter. The parameter is of „read only” type. Example:

```
cfg#uptime
Uptime 0d, 0:13:26
cfg#
```

The **"info"** command is used to display basic identification data of the module (including the inserted SIM). At the same time, information written in the QR code on the module label is displayed. This command is used only during initial setup and module diagnostics. Example:

```
cfg#info
-- Info --
Name : NB-R4B
SN : 75
Type : 850
SubType : 10

NB modem
IMEI : 864898061519593
SIM CCID : 89882390000727626486
SIM IMSI : 901288910195883
QRCODE :
https://mod.softlink.cz?type=850&subt=10&sn=0&imei=864898061519593&ccid=89882390000727626486
cfg#
```

3.1.6 Commands for setting communication with consumption meters and sensors

For setting the encoding system (communication protocol) and parameters of the RS-485 bus interface, there is a group of parameters listed in the configuration commands list in the **"All profiles"** sections. These are the following commands:

proto	<i>setting the communication protocol of the given input (OPTO/M-Bus/Modbus)</i>
ispeed	<i>initial communication speed of the bus interface</i>
parity	<i>setting the parity bit of serial communication (none/odd/even)</i>
periode	<i>setting the reading/transmission period</i>
irt	<i>setting the time interval for turning on the bus "rising time"</i>
ift	<i>setting the time interval for turning off the bus "falling time"</i>
iresp	<i>setting the timeout for response "response time"</i>
idel	<i>setting the minimum gap between commands "delay time"</i>
irep	<i>setting repeated reading</i>
iread	<i>command for immediate data reading</i>

These commands are set separately for each connected meter, so when entering them, it is always necessary to specify the meter's serial number ("index") as the first parameter.

Using the command "**proto** [index] [0/1/2]" we set the encoding system (communication protocol) for the given input, guided by which communication protocol the connected device on the given input communicates with. The NB-R4-V module allows setting these three communication protocols:

- IEC62056 protocol ("OPTO") - value "0"
- M-Bus protocol ("MBUS") - value "1"
- Modbus protocol ("MODBUS") - value "2"

The simplified designations used in parameter listings are shown in brackets. The current setting can be checked by entering the command without a variable.

Example of checking the current communication mode setting on the first input (index "0") and changing it to OPTO protocol:

```
cfg#proto 0
Protocol [0] is MODBUS
cfg#proto 0 0
Set Protocol[0] to OPTO with default values
cfg#
```

Example of changing the third input (index "2") to MBUS protocol:

```
cfg#proto 2 1
Set Protocol[2] to MBUS with default values
cfg#
```

Using the command "**ispeed** [index] [value]" we set the initial bit rate of the data interface. The module sends a data connection request to the connected device at this speed. Based on data exchange, the transmission speed can automatically increase to a value supported by the given type of meter (the connected device "agrees" with the module on a higher transmission speed).

Example of checking current values and subsequent setting of initial transmission speed for a meter with index "2":

```
mon#ispeed 2
Init speed [2] : 300 bps
mon#ispeed 2 600
Init speed [2] changed from 300 to 600 bps
cfg#
```

The command "**periode** [index] [value]" is used to set the reading period of the meter with the given index and send a message with the read values (the module sends the message immediately after reading). A different reading/sending period can be set for each of the six read devices (with index 0 to 5), when entering **value "0"** the given meter is **not read**. Zero value is set from the factory for all inputs.

The "periode" command **with index "6"** is used to set the transmission period of operational messages that the NB-R4-V module sends on its own behalf. The content of these messages is the operational data of the module (uptime, processor temperature, battery voltage...) and data from the history table (see description of the "hist" command). From the factory, the period for sending operational messages is also set to zero (transmission off).

Example of checking the setting of all transmission periods, setting the period of the first meter to 1 hour and subsequent checking of the transmission period of the first electricity meter:

```

mon#periode
Periode [0] is 0 min.
Periode [1] is 0 min.
...
Periode [5] is 0 min.
Periode [6] is 0 min.
mon#periode 0 60
Periode [0] changed from 0 to 60 min.
cfg#periode 0
Periode [0] is 60 min.

```

The commands **"irt"**, **"ift"**, **"iresp"**, **"idel"** and **"parity"** are used to set the parameters of serial data transmission via the data interface. They are set from the factory to suit the connection of devices commonly found on the market. It is recommended changing their settings only in specific cases, based on the documentation of the connected device. Parameter changes should always be made only by a qualified person with knowledge in the field of serial data transmission.

The **"irep [index] [value]"** command is used to set the number of attempts to read data from the given device. The value of this parameter is preset from the factory to "2", which means that if the data completeness check fails after the first data reading, the module will repeat the data reading attempt once more. This significantly increases the probability of obtaining a correct reading. Increasing the number of repetitions results in extending the bus activation time, which may have a slight effect on battery life. Example of checking current values and subsequent reduction of the number of attempts to read data for the meter with index "0":

```

mon#irep
Repeat[0] : 2
mon#irep 0 1
Repeat[0] changed from 2 to 1 * 50ms

```

The command **"iread [index]"** is used for immediate reading of current values from the connected device. Using this command, the functionality of the connection and reading of variables after connecting the meter to the module can be checked immediately.

Example of reading a message from a meter with index "0" with **M-Bus** encoding using the "iread" command:

```

cfg#iread 0
Reading configuration 0 ...
Reading mbus, deb 1...
  Enable uart on speed 2400 8E1
read 2
-- M-BUS long header --
Address   : 0
Ident     : 20020125
Manuf     : ACW
Version   : 14
Medium    : 7
Access    : 132
Status    : 0
Signature : 0x0000
DIF 0c, VIF 78, BCD8, val 20020126.000 'Fabrication no.', multi 0, save id 1
DIF 04, VIF 13, INT32, val 0.000 'm3', multi -3, save id 2
DIF 42, VIF 6c, INT16, val 14111.000 'date', multi 0, save id 4
DIF 44, VIF 13, INT32, val 0.000 'm3', multi -3, save id 3
DIF 04, VIF 6d, INT32, val 939592256.000 'date&time', multi 0, save id -1
DIF 02, VIF 27, INT16, val 355.000 'Operating time', multi 0, save id -1
  Found DIF/VIF 0c/78, val 20020126.000 Fabrication no.
  Found DIF/VIF 04/13, val 0.000 m3
  Found DIF/VIF 42/6c, val 14111.000 date
  Found DIF/VIF 44/13, val 0.000 m3
  Disable uart, end 60
mbus end 60

```

The result of reading a message from a connected meter with M-Bus encoding is displaying a summary of items from the M-Bus header and listing the current values of individual variables, including accompanying DIF/VIF data and their partial interpretation. At the end of each variable's record is information about which variable the given value was "mapped" to ("..save ID4" means that the given variable will be read as the fourth value, "..save ID -1" means that the module ignores the given variable). The values of variables that are set as read values var1, var2, var3 and var4 according to the procedure described in section 3.1.7 are listed again separately at the end of the listing so that it is visible whether the module really found all the set values in the message.

If it is not known what primary address the connected meter with M-Bus encoding has set, it could be discovered using a query with a general "broadcast" address as follows: - ensure that **only** the meter whose address we want to find out is connected to the RS-485 bus of the module,
- set the specially reserved address "254" for broadcast addressing on the appropriate internal port:

```
cfg#id 0 254
MBUS address [0] changed from 253 to 254
cfg#
```

- enter the "iread" command to read the M-Bus message of the connected device
- the primary address the device will display in the "Address" line.

Setting of read variables and their decoding data for the M-Bus protocol is described in detail in paragraph 3.1.7 "Commands for setting meters with M-Bus protocol".

Example of reading registers of a meter with index "1" with **OPTO** encoding using the "iread" command:

```
cfg#iread 1
Reading opto...
Enable uart on speed 4800
Send init id '' .. Recv 17 bytes : '/LGZ4ZMR120AR.510'
ack 4 (4800)
Set uart speed to 4800
: '/LGZ4ZMR120AR.510'
: 'F.F.0(00000000)'
: 'C.1.0(12420814)'
: 'C.90.1(11242814)'
: '1.8.1(000010.741*kWh)'
*Reg1 : '1.8.1' -> 1074
: '2.8.1(000000.000*kWh)'
: '21.8.0(000000.023*kWh)'
*Reg2 : '61.8.0' -> 37
: '22.8.0(000000.000*kWh)'
: 'C.7.1(0296)'
: 'C.7.2(0055)'
: 'C.7.3(0053)'
: 'C.7.0(0272)'
: 'C.8.0(00105143)'
: '! '
BCC 0x50 (0x50)
Flags 92
Recv end, 573 bytes
cfg#
```

As evident from the example, based on the "iread" command, the connected meter returns the current listing of its registers. The listing shows that:

- the read value reg1 (10.741 kWh) is stored in register "1.8.1"
- the read value reg2 (0.037 kWh) is stored in register "61.8.0"

These two values will be transmitted to the superior reading system in an INFO type message.

Setting of reading variables from specified registers for the IEC 62056 ("OPTO") protocol is described in detail in paragraph 3.1.8 "Commands for setting meters with OPTO protocol".

Example of reading registers of a meter with index "2" with **Modbus** encoding using the "iread" command:

```

cfg#iread 2
Reading configuration 2 ...
Reading modbus...
  Enable uart on speed 9600
Modbus send  : (8 bytes):
01 03 01 c1 00 03 55 cb
Modbus recv  : (11 bytes):
01 03 06 00 00 00 06 4f 63 21
  Address : 1
  Register : 450 (0x01c2)
  Value (INT48) : 1615
  Read address register ...
Modbus send  : (8 bytes):
01 03 00 04 00 04 05 c8
Modbus recv  : (13 bytes):
01 03 08 00 00 00 00 17 11 27 24 db 8d
  Address : 1
  Register : 5 (0x0005)
  Value (BCD16) : 17112724
  Device address : 17112724
cfg#

```

As evident from the example, based on the "iread" command, the connected meter returns the current listing of two set registers ("450" and "5"). The listing shows that:

- the meter has address "1" set on the Modbus bus
- the meter's serial number (17112724) is read from register "5"
- the required value (currently "1615") is read from register "450"
- no other value is set

The two values are read into the output WMBUS type message: the meter's serial number and the value of register "450".

Setting of reading variables from specified registers for the Modbus protocol is described in detail in paragraph 3.1.9 "Commands for setting meters with Modbus protocol".

3.1.7 Commands for setting M-Bus meters

This section contains commands for setting internal inputs of the NB-R4-V module for meters with M-Bus encoding. The commands are listed in the HELP section "Wired MBUS commands per meter [0 - 5]" and are always entered with the meter index, i.e. in the form "[command] [index] [value]". Up to four measured variables can be read from devices with the M-Bus protocol. These are the following commands:

id	<i>setting the primary address of the meter according to the M-Bus standard (number from 0 to 255)</i>
sid	<i>setting the secondary address of the meter according to the M-Bus standard (number from 0 to 99999999)</i>
mcount	<i>setting the maximum number of read M-Bus packets (0 - 10)</i>
var1	<i>setting the "DIF VIF" values for selecting the first variable</i>
var2	<i>setting the "DIF VIF" values for selecting the second variable</i>
var3	<i>setting the "DIF VIF" values for selecting the third variable</i>
var4	<i>setting the "DIF VIF" values for selecting the fourth variable</i>

Setting primary and secondary M-BUS address

The "id" variable is used to set the **primary** ("bus") address of the connected meter according to the M-Bus standard. This identifier is used for addressing messages between the NB-R4-V module and the connected meter, so it practically determines which meter is connected to which internal input of the module. The currently set primary address can be displayed using the command in the format "**id [index]**" (without parameter). Change the identifier by entering the desired primary M-Bus address after the "id" command and index, which must be from the range of numbers 0 to 255.

Example of setting the primary M-Bus address of the meter with index "0" and subsequent checking of the setting:

```
cfg#id 0
MBUS address [0] : 254
cfg#id 0 126
MBUS address [0] changed from 254 to 126
cfg#
```

The **"sid"** variable is used to set the **secondary** ("individual") address of the connected meter according to the M-Bus standard (usually corresponds to the device's serial number). This identifier is used for addressing messages between the NB-R4-V module and the connected meter, so it practically determines which meter is connected to which internal input of the module. The currently set secondary address can be displayed using the command in the format **"sid [index]"** (without parameter). Change the identifier by entering the desired secondary M-Bus address after the **"sid"** command and index, which must be from the range of numbers 0 to 99999999.

Example of displaying the current setting of the secondary address (serial number) and its subsequent setting to the value "12459832":

```
cfg#sid 0
MBUS secondary address [0] :
cfg#sid 0 12459832
MBUS secondary address [0] changed from  to 12459832
cfg#
```

Note: When querying M-Bus devices, either the primary (bus) address or the secondary (individual) address can be used. When query with using of the secondary address, the primary address must always be set to the specially reserved value "253". When setting the secondary address using the above procedure, the **primary address automatically changes to the value "253"**. Example:

```
cfg#id
MBUS address [0] : 111
MBUS address [1] : 126
cfg#sid 0 33221221
MBUS sec. address [0] changed from 0 to 33221221
cfg#id
MBUS address [0] : 253
MBUS address [1] : 126
cfg#
```

The setting of communication using primary or secondary address is evident from the configuration listing of the given module:

```
----- Configuration 0 -----
  MBUS mode
  Uart speed 2400 8E1
  Meter address : secondary 33221221
  MBUS max more packets : 0
  ...
----- Configuration 1 -----
  MBUS mode
  Uart speed 2400 8E1
  Meter address : 126
  MBUS max more packets : 0
```

From the listing, it is clear that the module communicates with the device with index "0" via the secondary address, and with the device with index "1" via the primary address.

For displaying both primary and secondary addresses in bulk use the **"id"** and **"sid"** commands without an index.

```

cfg#id
MBUS address [0] : 253
MBUS address [1] : 126
cfg#sid
MBUS sec. address [0] : 33221221
MBUS sec. address [1] : 00000000

```

The M-Bus protocol packet has a length limited to 255 Bytes. If a device with the M-Bus communication protocol offers so many variables that they do not fit into one M-Bus packet, the packet is marked with a "More Packets" flag, which signals that the continuation of the message is in the next packet. If a "Master" type device detects this flag, it can request more and more packets until it reads the entire message.

Using the command "**mc**ount [index] [number]", the **maximum number of read packets** can be set, so that packets in which there is no required variable are dropped. For example, if given device has all the required variables in the first two packets, it is useful to set the maximum number of "additional" packets to "1", which means that the module will read only the first two packets. During manufacturing, the default value is set to "0" (no additional packets), the current setting value can be displayed using the command "**mc**ount [index]" (without parameter). The maximum value that can be entered is "10", which means that the module will read 11 M-Bus packets.

Example of checking the current setting and changing the setting of maximum number of read packets to "1":

```

cfg#mcount 0
Max more packets [0] : 0
cfg#mcount 0 1
Max more packets [0] changed from 0 to 1
cfg#

```

Setting the selection of required variables

The NB-R4-V module can read **up to four variables** from each read device. The message in M-Bus format from the given device type may contain many different variables, from which it is necessary to select those variables whose values will be transmitted in INFO messages to the superior system using the commands "var1", "var2", "var3" and "var4".

The selection is made by setting the values of DIF ("DIF" = Data Information Field, sometimes also "DIB" - Data Information Block) and VIF ("VIF" = Value Information Field, or also "VIB" = Value Information Block), which are always unique in the message for a specific variable. So when (for example) the DIF/VIF value is set to "02 5B", the system will select from the WMBUS message the value of the variable that is marked in the message by this combination of DIF/VIF.

*The **DIF/DIB** value expresses the data type (Integer, BCD...), general character (minimum/maximum/average value) and if there are more values of the given type, also the source number ("Storage Number"). The **VIF/VIB** value specifies the type of measured quantity (amount of energy, power, volume, flow, temperature...), unit of measurement (mV, kWh, m3...) and multiplier. The method of determining the DIF/VIF decoding data is clearly defined by the EN 13757-3 standard, which allows correct decoding and interpretation of data from the M-Bus message even if the message comes from an unknown device for which the documentation is not available.*

Using the command "**var**1 [index] [DIF] [VIF]", the DIF and VIF values of the first read variable will be set. The module will select from the M-Bus message of the connected device the value for which these accompanying data are present. Similarly, by using the "var2", "var3" and "var4" commands, the DIF/VIF parameters for all required variables will be preset. The DIF/VIF values are always separated by a space and are entered either in hexadecimal format (with the "0x" prefix) or in decimal format. In the listing, the values are always displayed in hexadecimal format.

Example of setting all four read variables for a water meter with M-Bus output by entering DIF/VIF values in hexadecimal format:

```

cfg#var1 0 0x0c 0x78
DIF/VIF [0/1] : 0c 78
cfg#var2 0 0x04 0x13
DIF/VIF [0/2] : 04 13
cfg#var3 0 0x44 0x13
DIF/VIF [0/3] : 44 13
cfg#var4 0 0x42 0x6c
DIF/VIF [0/4] : 42 6c

```

Example of an alternative way to set the "var4" variable by entering DIF/VIF values in decimal format:

```
cfg#
cfg#var4 0 66 108
DIF/VIF [0/4] : 42 6c
cfg#
```

Correctness of the previous settings can be checked with using the "iread" command, where the "mapping" of DIF/VIF codes to var1 to var4 is displayed for the selected variables. Besides, all found values are listed at the end of the listing (see the description of the "iread" command in paragraph 3.1.6 "Commands for setting communication with consumption meters and sensors"). For the above set values, the last part of the "iread" listing looks like this:

```
Found DIF/VIF 0c/78, val 20020126.000 Fabrication no.
Found DIF/VIF 04/13, val 0.000 m3
Found DIF/VIF 42/6c, val 14111.000 date
Found DIF/VIF 44/13, val 0.000 m3
```

It is evident from the listing that the module found in the M-Bus message current values for all set variables. The DIF/VIF values are set to "00 00" by default for all four variables.

3.1.8 Commands for setting IEC 62056 ("OPTO") meters

This section contains commands for setting internal inputs for reading meters with IEC 62056 ("OPTO") encoding connected to the NB-R4-V module. The commands are listed in the HELP section "Opto protocol commands per meter [0 - 5]" and are always entered with the meter index, i.e. in the form "[command] [index] [value]". Up to four measured variables can be read from devices with the OPTO protocol. These are the following commands:

oid	<i>setting the meter identifier on the bus</i>
mspeed	<i>setting the maximum communication speed of the interface</i>
reg1	<i>setting the register address for variable "1"</i>
reg2	<i>setting the register address for variable "2"</i>
reg3	<i>setting the register address for variable "3"</i>
reg4	<i>setting the register address for variable "4"</i>

The "**oid [index] [value]**" command is used to set a unique bus identifier (OID) for the given index of the meter according to the IEC 62056-21 standard. This command associates the index of the connected device from the range (0 to 5) in the configuration of the NB-R4-V module with a specific device. The module uses the OID identifier when querying so that it addresses the query to a specific meter and receives only one response (from the queried meter).

IMPORTANT NOTE! If the OID value is not set for a given index, the module queries using the broadcast address and stores the response it receives. If multiple meters are connected to the module, the module cannot distinguish which one the response came from. Therefore, if multiple meters are connected to the NB-R4-V module and bus identifiers are not set for all of them, the data cannot be read. **In case the bus identifiers are unknown, or the connected device (meter) does not respond to them (i.e. they are not stored in the meter configuration), only one meter can be connected to the NB-R4-V module.**

The bus identifier can be found from the meter documentation or by querying its manufacturer. It is often identical to the serial number, or it is a designated part of the serial number (but this is not a rule). In the above example of register listing the "oid" value (837224) is stored in register C.90, but for other types of devices it may be in a different register (or none).

Example of setting a unique bus identifier for a meter to index "1" and checking data reading using OID:


```

cfg#oid 1 837224
Meter ID [1] changed from  to 837224
cfg#iread 1
Reading configuration 1 ...
Reading opto...
  Enable uart on speed 300 7E1
  Send init id '837224' .. Recv 18 bytes : '/ZPA4ZE110.v30_012'
  ack 4 (4800)
  set 4 (4800)
  Set uart speed to 4800
: 'F.F(000000)'
: 'C.1.0(05837224)'
*Mid : 05837224
: 'C.90(837224)'
: '1.8.1(0000008.9#kWh)'
*Reg1 : '1.8.1' -> 8.900
: '2.8.1(0000000.0#kWh)'
. . .

```

From the example, it is clear that the OID is set correctly because the electricity meter responds to a specific query using the entered OID.

Using the command **"mspeed [index] [value]"** the automatic increase of the bus speed can be limited to the preset value (i.e. to a lower speed than what the connected device allows) so that the reading is more reliable. Example of checking the current value and then setting the maximum transmission speed for the meter with index "2":

```

cfg#mspeed 2
Max speed [2] : 4800 bps
cfg#mspeed 2 9600
Max speed [2] changed from 4800 to 9600 bps
cfg#

```

The commands **"reg1"**, **"reg2"**, **"reg3"** and **"reg4"** are used to set the designations (also called "addresses" or "OBIS-codes") of the registers from which the required variables will be read. The designation of registers in the field of electricity measurement is described in the IEC 62056-61 standard, so when reading electricity meters, the setting of registers is unambiguous even without documentation for the given electricity meter. The register addresses are preset from the factory according to the conditions of CEZ-Distribution company (major Czech electricity distributor):

- Reg1: 'C.1.0' - "Meter Serial Number"
- Reg2: '1.8.1' - "Positive active energy (A+) in tariff T1 [kWh]"
- Reg3: '1.8.2' - "Positive active energy (A+) in tariff T2 [kWh]"
- Reg4: '2.8.0' - "Negative active energy (A+) total [kWh]"

Warning. When reading electricity meters of the electricity distributor, it is necessary to follow its conditions and possibly also the conditions of the market regulator. These conditions may include a list of registers that a third party is allowed to read. Setting the reading of an unauthorized register may result in disconnection of the measurement or another type of reaction/sanction from the distributor.

The command for setting the read register **"reg1"** has the format **"reg1 [index] [value]"**, where the value is the register designation in text format (for example "C.1.0" or "1.8.1"). Setting the other three registers is done in a similar way using the commands **"reg2"**, **"reg3"** and **"reg4"**. Setting of the register can be cleared by entering the value "." (dot).

The settings of the register reading can be checked by entering the **"regX"** command with an index, without a value. Example:

```

cfg#reg1 0
Reg1 [0] : 'C.1.0'
cfg#reg2 0
Reg2 [0] : '1.8.1'

```

Example of changing the setting of read registers (values 1.8.1, 1.8.2 and 4.8.0 will be read, the fourth value will be

excluded from reading):

```
cfg#reg1 0 1.8.1
Reg1 [0] changed from 'C.1.0' to '1.8.1'
cfg#reg2 0 1.8.2
Reg2 [0] changed from '1.8.1' to '1.8.2'
cfg#reg3 0 4.8.0
Reg3 [0] changed from '1.8.2' to '4.8.0'
cfg#reg4 0 .
Reg4 [0] changed from '2.8.0' to ''
cfg#
```

The set values are displayed in the "iread" listing, where the selected variables are marked with an asterisk (see the description of the "iread" command in paragraph 3.1.6 "Commands for setting communication with consumption meters and sensors").

3.1.9 Commands for setting meters with Modbus protocol

This section contains commands for setting internal inputs for reading meters with Modbus protocol encoding connected to the NB-R4-V module. The commands are listed in the HELP section "ModBus protocol commands per meter [0 - 5]" and are always entered with the meter index, i.e. in the form "**command [index] [value]**". Up to four measured variables can be read from devices with the Modbus protocol. These are the following commands:

id	<i>setting the meter identifier on the Modbus bus (0 - 255)</i>
reg0	<i>setting the starting address of the register for reading the device serial number</i>
type0	<i>setting the data type of the register for reading the device serial number</i>
func0	<i>setting the type of Modbus command for reading the register for the device serial number</i>
reg1	<i>setting the starting address of the register for the first variable</i>
type1	<i>setting the data type of the register for the first variable</i>
func1	<i>setting the type of Modbus command for reading the register for the first variable</i>
reg2	<i>setting the starting address of the register for the second variable</i>
type2	<i>setting the data type of the register for the second variable</i>
func2	<i>setting the type of Modbus command for reading the register for the second variable</i>
reg3	<i>setting the starting address of the register for the third variable</i>
type3	<i>setting the data type of the register for the third variable</i>
func3	<i>setting the type of Modbus command for reading the register for the third variable</i>
reg4	<i>setting the starting address of the register for the fourth variable</i>
type4	<i>setting the data type of the register for the fourth variable</i>
func4	<i>setting the type of Modbus command for reading the register for the fourth variable</i>
sc	<i>sending a command to read the content of the specified register</i>

The module supports reading the serial number of the connected device and **up to 4 register values** and sending all read values in an INFO message. The commands "reg0", "type0" and "func0" are used to read the serial number of the connected device. For reading values 1, 2, 3 and 4, the commands "reg1", "type1", "func1", "reg2", "type2", "func2", "reg3", "type3", "func3" and "reg4", "type4", "func4" are used similarly.

The "**id**" command is used to **set the identifier (address) of the connected meter** according to the Modbus standard. This identifier is used for addressing messages between the NB-R4-V module and the connected meters. The currently set identifier can be displayed using the command in the format "**id [index]**" (without parameter). The identifier can be changed by entering a number from the range 1 to 247 after the "id" command and index (address "0" is reserved for broadcast, addresses "248" to "255" are in reserve).

Example of setting the identifier of the meter with index "0" to the value "5" and subsequent checking of the current setting:

```
cfg#id 0 5
ModBus address [0] changed from 1 to 5
cfg#id 0
ModBus address [0] : 5
cfg#
```

Setting registers for reading variables

The commands **"reg"**, **"type"** and **"func"** are used in the Modbus system to set the register address and form the command for reading it. The commands **"reg0"**, **"type0"** and **"func0"** are used only to set the Modbus command for reading the register where the device serial number is stored.

Using the command **"reg0 [index] [value]"**, select the **starting address of the Modbus register** where the device serial number information is located. Refer to the meter documentation for the starting address of the register (*) containing the serial number.

() The Modbus system stores data in 16-bit registers ("words"). If it is necessary to store information containing more than 16 bits (= 2 Bytes), multiple consecutive registers are used. To read the data, it is therefore necessary to enter the starting address of the register and the number of consecutive registers (words) to be read (see the "type" command).*

The **"reg0"** register is not preset from the factory, it has a default value of **"0"**. Example of setting the **"reg0"** register for reading the serial number of the meter with index **"0"** to address **"5"**:

```
cfg#reg0 0 5
Reg0 [0] changed from 0 to 5
cfg#
```

Using the command **"type0 [index] [value]"**, the **data format of the register** can be set. This parameter determines how many consecutive registers ("words") should be read and in what format the numbers are encoded (Integer, Binary Code Decimal, Float, Double Precision Float, Unsigned Integer). The **"floatH"** data format is intended for reading data in **"float"** format with reverse byte order (from most significant to least significant).

The command is entered by setting a number from 0 to 82, where each number represents one preset variant. The following variants are available:

- 0 - NONE (the given register is not read - "disabling" the variable)
- 1 - INT8 (1 word)
- 2 - INT16 (1 word)
- 3 - INT32 (2 words)
- 4 - INT48 (3 words)
- 5 - INT64 (4 words)
- 6 - float (2 words)
- 7 - double (4 words)
- 8 - BCD2 (1 word)
- 9 - BCD4 (1 word)
- 10 - BCD8 (2 words)
- 11 - BCD12 (3 words)
- 12 - BCD16 (4 words)
- 13 - floatH (2 words)
- 14 - UINT8 (1 word)
- 15 - UINT16 (1 word)
- 16 - UINT32 (2 words)
- 17 - UINT48 (3 words)
- 18 - UINT64 (4 words)

For each variant, the data type abbreviation is given and the number of read "words" is shown in parentheses.

We can display the data type variants using the command **"typeX ?"**. Example:

```

cfg#type1 ?
Modbus types :
NONE
INT8
INT16
INT32
INT48
INT64
FLOAT
DOUBLE
BCD2
BCD4
BCD8
BCD12
BCD16
FLOATLH
UINT8
UINT16
UINT32
UINT48
UINT64

```

When setting each value, it must be taken into account in which format and in how many "words" the value is stored. For example, if it is a device serial number in four "words", a typical format setting for reading it is "BCD16". Example of setting the "type0" register format to value "12" (BCD16):

```

cfg#type0 0 12
Type0 [0] changed from 10 to 12 (BCD16)
cfg#

```

The command "**func0 [index] [value]**" can be used to select the function of the Modbus protocol, which will be used to read the required register. The following four functions are available, numbered 1 to 4:

- 1 - reading a series of binary information of the "Coils" type (typically outputs of binary sensors)
- 2 - reading a series of binary information of the "Discrete Inputs" type (typically settable binary values 0/1)
- 3 - reading a series of 16-bit registers "Holding Registers" (typically settable parameters)
- 4 - reading a series of 16-bit registers "Input Registers" (typically analog "read only" inputs)

When choosing a function, be guided by the type of register in which the read variable is stored ("Coil", "Discrete Input", "Input Register", or "Holding Register") and whether the registers are divided by types into separate blocks requiring a read function of the given block (see the description of the Modbus protocol - "Modbus Application Protocol Specification" at www.modbus.org).

*Functions 1 and 2 are intended for reading binary registers. Function 4 is intended for reading registers of the "Input Registers" type, which are rarely used in new devices. For "consumption meter" type devices, **in most cases, registers of the "Holding Register" type are read using function 3, which is set by default for all variables.***

Example of setting Modbus function number "3" for reading the "reg0" register for the meter with index "0":

```

cfg#func0 0 3
Func0 [0] changed from 1 to 3
cfg#

```

The functions "reg1", "type1", "func1", "reg2", "type2", "func2", "reg3", "type3", "func3", "reg4", "type4" and "func4" are used similarly to set the reading of variables 1 to 4. A maximum of 4 variables can be read and transferred (for an electricity meter, these could be, for example, the states of counters of two tariffs of active energy and the states of counters of reactive energy Qi and Qc). When transfer a smaller number of variables (for example, only readings of two tariffs of active energy), the remaining variables can be "turn off" by setting the "type" parameter to value "0" (NONE).

The settings of the above described trio of parameters ("reg", "type" and "func") can be checked by performing a control reading of the register using the "iread" command described in section 3.1.6 "Commands for setting communication with consumption meters and sensors".

Bulk listings of parameter settings for reading variables can be displayed with a command without an index - see example:

```
cfg#type1
Type1 [0] is INT48
Type1 [1] is INT48
Type1 [2] is INT48
Type1 [3] is INT48
Type1 [4] is INT48
Type1 [5] is INT48
cfg#
```

The following **default start addresses** registers are set by default for the Modbus communication protocol:

```
Reg0 - value "0" (disabled)
Reg1 - value "450"
Reg2 - value "514"
Reg3 - value "578"
Reg4 - value "642"
```

The default setting of the "Type" value is "INT48" for all variables, the default setting of the "Function" value is "3" for all variables.

The "sc" command can be used to **display the value of any register** of a connected Modbus device to the command line. This command can be used for verifying the correctness of reading required register before setting individual parameters into the profile. The command has the following syntax:

```
"sc [index] [function] [starting register address] [number of register words]"
```

To execute the command, at least the network address of the read device must be entered into the profile.

3.1.10 Commands for setting communication with the NB-IoT network

This group of commands is used to set up the message sending system. These are the following commands:

server	<i>setting the IP address of the target server</i>
sport	<i>setting the port number of the target server</i>
testip	<i>setting the IP address for control ping</i>
sreply	<i>redirecting the response to the target server</i>
apn	<i>setting the name of the private network access point (Access Point Name)</i>
sess	<i>maximum time for establishing a connection with the server</i>
errtime	<i>time interval for modem restart after entering an error state</i>
band	<i>setting the NB frequency band (default "20" = Europe)</i>
tconn	<i>maximum waiting time for a response from the server</i>
sping	<i>sending a control "ping" to the specified address</i>

The module sends messages wrapped in UDP packets of the Internet protocol to a preset **target server**, on which the remote data collection application is running. The following commands are used to **set the IP address and target port number** and to set the **name of the communication gateway** between the GSM operator's network and the Internet (so-called "APN" = Access Point Name).

Using the "server" command, we set the **IP address of the target server**. The address is entered in decimal format in the commonly used way.

Example of setting the target server IP address to "92.89.162.105" and checking the setting:

```
cfg#server 92.89.162.105
Server changed from '0.0.0.0' to '92.89.162.105'
cfg#
cfg#server
Server is : '92.89.162.105'
cfg#
```

Using the "sport" command, we set the **UDP port number** of the target server that corresponds to the remote data collection application. Example of setting the target server UDP port number to "2000" and checking the setting:

```
cfg#sport 2000
UDP port changed from 0 to 2000
cfg#sport
UDP port : 2000
cfg#
```

The **"sreply"** command is used to specify the setting of **communication via the reverse channel** (see paragraph 3.3 "Setting module parameters from a remote computer using the reverse channel"). In some NB-IoT networks/services, it is possible to send return messages to the module only from a different IP address than the standard set IP address of the target server for sending messages. When the module is set to "Reply to server : no", the module responds to messages in a way that is standard for IP networks - i.e., it responds to the address from which the query came. When set to "Reply to server : yes", the module always responds to queries from any server to the set target server address (see the "server" command). We set the module to the "yes" state by entering parameter "1", we set the module to the "no" state by parameter "0".

Example of checking the reverse channel communication setting and then making a change:

```
cfg#sreply
Reply to server : no
cfg#sreply 1
Reply to server : yes
cfg#
```

If the GSM network operator transmits data from modules to their operator in the form of a virtual network, we use the **"apn"** command to set the name of the communication gateway between the GSM network and the Internet (so-called "APN" = "Access Point Name"), reserved for the given virtual network within the GSM network. The APN name is assigned to virtual network operators by the GSM network operator. We cancel the APN setting by entering the value "." (dot).

Example of setting the APN name to "cms.softlink":

```
cfg#apn cms.softlink
APN changed from '' to 'cms.softlink'
cfg#apn
APN is : 'cms.softlink'
cfg#
```

The current setting of the target server and communication gateway is displayed in the configuration listing as follows:

```
Server IP : '92.89.162.105'
Server port : 2000
My src port : 2000
APN : 'cms.softlink'
```

The "My src port" value is the UDP port number of the module itself. This value is "read only" and cannot be changed.

Using the **"sess"** command, we set the **maximum time for establishing a connection with the operator's server ("session time")** in minutes. Some GSM service operators charge for each connection establishment ("session"), so establishing a connection before sending each message can be financially disadvantageous (and sending a message takes longer). On the other hand, if the server loses the connection during a permanent connection for some reason, the module does not receive any message about it from the network and the sent messages are lost. The "sess" parameter can be used to set a time after which the module checks the functionality of the connection using the "ping" function (see the use of the "testip" command below). If the functionality of the connection is not verified using the control "ping", the module terminates the connection and establishes it again when sending the next data. By default, this time is set to **2 days** (172800 seconds, 2880 minutes), which is a reasonable compromise between costs and message delivery reliability. If the GSM operator does not charge for connection establishment, the parameter can be set to a shorter time (or even to zero, when a connection is established when sending each message), but for the sake of shortening the communication time, we recommend keeping the default setting even in this case.

The current setting of the maximum connection establishment time is displayed in the configuration listing as

follows:

```
Max session time 172800 sec - 2d, 0:00:00
```

Example of setting the maximum connection establishment time to 2880 minutes:

```
cfg#sess 2880
Max session time : 2880 min.
cfg#
```

Using the **"testip"** command, we set the **IP address for the control ping**. The address is entered in decimal format in the commonly used way. The control ping is sent after the end of the maximum time for establishing a connection with the operator's server (see the previous "sess" parameter). The control ping is addressed to the set address of a suitable computer in an accessible IP network (a computer that reliably responds to control "ping" queries). If a response comes to the ping, the connection with the NB-IoT network is verified and it is not necessary to establish it again.

Example of setting the IP address of the computer for sending a control "ping" to the value "10.0.0.1":

```
mon#testip 10.0.0.1
Test ip changed from '10.0.0.8' to '10.0.0.1'
mon#
```

We can check the availability of the server for the control "ping" message using the **"sping [address]"** command. By entering this command, the system sends a control ping and displays the result.

Using the **"tconn"** command, we set the **maximum waiting time for a network response when establishing a connection**. If the GSM operator's network does not respond to connection requests within this time, the module's GSM modem turns off and attempts to establish a connection when sending the next message. The parameter is set by default to **5 minutes** (300 seconds). We recommend changing the value if the GSM network operator guarantees a significantly different response from the network. Example of changing the setting of the maximum waiting time for a network response when establishing a connection from 200 to 300 seconds (5 minutes):

```
mon#tconn
Connection timeout is 200 sec
mon#tconn 300
Connection timeout is 300 sec
```

*Both of the above parameters ("sess" and "tconn") affect power consumption and **battery life**. If a connection is established with the server when sending each message, the active state of the modem is prolonged, when it consumes a lot of energy. If too long a waiting time for network response ("tconn") is set, the modem is unnecessarily on for a long time while waiting for connection establishment. From this point of view, it is advantageous to set the longest possible "sess" time and the shortest possible "tconn" time. However, such a setting **reduces the reliability of message delivery**, because in the event of a "session" failure on the operator's side, messages are lost until the "sess" time expires, and with a short "tconn" timeout, it may happen that the connection is not established in time and the message is not sent. The setting of both parameters must be a compromise between energy efficiency and message delivery reliability.*

Using the **"errtime"** command, we set the **modem restart time after an error in establishing a connection**. If a fatal error occurs when attempting to establish a connection, the modem automatically deactivates to avoid unnecessarily draining the battery with repeated connection attempts. The "errtime" parameter sets a timer that restarts the modem after the set time and attempts to establish a connection again. The parameter is set by default to **24 hours** and we recommend not changing this value without reason. Example of checking the "errtime" parameter setting and changing its value to 48 hours:

```
mon#errtime
Error restart time : 24 hours
mon#errtime 48
Error restart time : 48 hours
cfg#
```

Using the **"band"** command, you can set the **NB-IoT modem frequency band**. By default, the most commonly used frequency band B20 in Europe is set (value "20"). The used modem may support multiple frequency bands,

in which case it is possible to switch the module to another frequency band. In different production series of the NB-R4-V module, the used modem modification may differ depending on the current availability and price of the modem at the time of production. **If you are interested in using the modem in a band other than B20 (800 MHz), always include this information in the order, or contact the manufacturer.**

In the "Narrow band" section of the "HELP" listing, the commands "tshort", "tlong" and "at" are also displayed, which are used exclusively for initial setup and module diagnostics. **We strongly recommend against using these commands during device operation.**

3.1.11 Commands of the "Utils" group for setting and checking basic module functions

This group of commands is used to set the content and period of message sending and other common module functions. The following commands are used to set the content and period of sending information messages:

periode	<i>setting the period of spontaneous message sending</i>
ekey	<i>setting the encryption key (". " - encryption disabled)</i>
hist	<i>setting the data reading period in "history" mode</i>
hdata	<i>preview of the current content of the historical readings table</i>

The "**Periode**" variable is used to set the period of spontaneous sending of information messages. For the NB-R4-V module, this parameter is always set with indices 0 to 6, which have the following meanings:

- values **with indices 0 to 5** are used to set message sending **from individual electricity meters** in "online" mode, where each electricity meter is read with its own period and the message is sent immediately;
- the value **with index "6"** is used to set message sending **from the module itself**. These messages have their own repeat period and carry service information related to the module. In "**history**" mode (see description of the "hist" variable below), these messages also carry tables of previously measured values from individual electricity meters.

During production, this parameter is set to "0" for all 6 indices, which turns off transmission. We set the transmission period for a given index by entering the transmission period in minutes as a parameter of the command (theoretically up to 65535 minutes can be set). The "**periode**" command (without a parameter) can be used to display the current setting values.

Example of displaying currently set values of the "periode" parameter for all indices:

```
cfg#periode
Periode[0] is 60 min.
Periode[1] is 120 min.
Periode[2] is 120 min.
Periode[3] is 0 min.
Periode[4] is 0 min.
Periode[5] is 0 min.
Periode[6] is 720 min.
cfg#
```

Example of setting the transmission period for electricity meter with index "2" to 60 minutes and shortening the transmission period of the module's service message to 360 minutes:

```
cfg#periode 2 60
Periode[2] changed from 120 to 60 min.
cfg#periode 6 360
Periode[6] changed from 720 to 360 min.
cfg#
```

With this setting, the electricity meter with index 2 will be read every 60 minutes and the message with its readings will be sent immediately. Every 6 hours, a message with the module's operational parameters will be sent.

Warning: NB-IoT service operators may charge for this service based on the volume of data transferred. Higher transmission frequency negatively affects battery life and may also negatively affect the cost of the service.

The "**Encryption code**" variable is used to set the encryption key for encrypting outgoing module messages using the AES-128 key. We enter the 16-byte encryption key using the "**ekey**" command followed by a string of 16 bytes, which can be entered in decimal or hexadecimal form (see examples).

Example of entering the encryption key in hexadecimal form:

```
cfg#ekey 0x1a 0x2b 0x3c 0x4d 0x5e 0x6f 0xa1 0xb2 0xc3 0xd4 0xe5 0xf6 0x77 0x88 0x99 0xaf
Setting encryption key : 1a 2b 3c 4d 5e 6f a1 b2 c3 d4 e5 f6 77 88 99 af
```

Example of entering the encryption key in decimal form:

```
cfg#ekey42 53 159 188 255 138 241 202 136 21 98 147 235 15 145 136
Setting encryption key : 2a 35 9f bc ff 8a f1 ca 88 15 62 93 eb 0f 91 88
```

After entering the encryption key, the information about encryption being turned on *”Data will be encrypted by AES”* will be displayed in the list of set parameters (see paragraph 3.1.1).

We turn off encryption by entering the parameter *”.”* (dot) after the *”ekey”* command:

```
cfg#ekey.
Encryption disabling
```

After turning off encryption, the information *”Data will be unencrypted”* will be displayed in the parameter listing (see paragraph 3.1.1).

To reduce the number of transmissions (saving battery capacity), the NB-R4-V module allows sending a larger number of previously read values in one message. Such a message then does not contain current measured values, but a set of previously measured values stored in the internal memory of the module (hereinafter *”historical readings”*). Each set of historical data contains data from all read electricity meters that are not read in *”online”* mode, but in **”history” mode**. Each set of historical readings is also assigned the time of their acquisition, which is also transmitted to the central system. To determine the number of transmitted sets of historical data, it is necessary to take into account these limitations:

1. The module’s memory size allows storage of **up to 100 historical readings**. The number of historical receive windows that can be transmitted in one transmission session depends on the number of electricity meters read and the number of variables read. With the standard setting, 4 data are read from each electricity meter: serial number, T1 reading, T2 reading and counter reading of supply to the network (in case of production). If we read data in *”history”* mode from only one electricity meter, we can stack up to 25 readings in memory, which allows, for example, reading the electricity meter every 15 minutes and sending all readings at once in one message sent with a period of 6 hours. After each message is sent, the historical readings table is emptied
2. The size of the NB IoT data message is limited to 512 kB, so data from about 10 variables fit into one IoT data message. If a message with a larger amount of historical data is transmitted, the module divides the transmitted data into several NB-IoT messages. This may affect the charging of the service.

From these dependencies, it follows that setting the reading period and data transmission period when reading a larger number of electricity meters is **always a compromise between information delay, energy consumption and service cost**. To minimize information delay, it is necessary to transmit as often as possible. To minimize energy consumption, on the other hand, it is advantageous to transmit as infrequently as possible. To minimize cost, it is advantageous to fill the transmitted packets as much as possible.

Determining the parameters of measurement and transmission frequency should always be done on a project basis, taking into account the specific situation, specific tariff and specific needs and requirements of the project.

Example: *If the total transmission period of the module (see description of the *”periode”* parameter with index *”6”*) is set to 240 minutes (4 hours), in *”history”* mode 3 electricity meters are read with a reading period of 30 minutes, each electricity meter generates $240/30 = 8$ sets of data over the entire transmission period, which is a total of $8*4 = 32$ historical readings. All three electricity meters therefore generate a total of $3*32 = 96$ historical readings, which does not exceed the maximum capacity of the history memory (100 readings).*

We set the **reading period with storing readings in memory** for a given electricity meter using the **”hist [index]”** command. The value is set in minutes, allowed setting values are 10, 15, 30 and 60 minutes (if another number is entered, the nearest of these values is set). When set to *”0”* (default setting), readings are not stored in memory. Example of setting the period for storing readings for electricity meters with indices *”0”* and *”1”* with periods of 15 and 30 minutes:

```
mon#hist 0 15
History[0] changed from 0 to 15 min.
cfg#hist 1 30
History[1] changed from 0 to 30 min.
cfg#
```

One NB-R4-V module can simultaneously connect electricity meters with reading in "online" mode and in "history" mode. If we set the value "periode a x" for the electricity meter with index [a], it is read in "online" mode with period "x". If we set the value "hist b y" for the electricity meter with index [b], it is read in "history" mode with period "y". When making changes, the last setting always applies, so if a new "hist" period is set for an electricity meter with "online" reading, its reading method switches to "history" (...and vice versa). The valid reading method is displayed in the configuration listing, where there is always either the "Send periode" data (for "online" mode) or the "Send history" data (for "history" mode).

Example of a module configuration listing with two electricity meters in "history" mode, two electricity meters in "online" mode and two unoccupied inputs:

```
---- Configuration 0 -----
  Send history : 15 min.
---- Configuration 1 -----
  Send history : 30 min.
---- Configuration 2 -----
  Send periode : 120 min.
---- Configuration 3 -----
  Send periode : 60 min.
---- Configuration 4 -----
  Send periode : 0 min.
---- Configuration 5 -----
  Send periode : 0 min.
```

When listing the settings of all indices, the same situation will be displayed as follows:

```
cfg#periode
Periode[0] is - min.
Periode[1] is - min.
Periode[2] is 120 min.
Periode[3] is 60 min.
Periode[4] is 0 min.
Periode[5] is 0 min.
Periode[6] is 1440 min.
```

```
cfg#history
History[0] is 15min.
History[1] is 30 min.
History[2] is 0 min.
History[3] is 0 min.
History[4] is 0 min.
History[5] is 0 min.
cfg#
```

In "history" mode, the module records data from electricity meters with higher frequency, but sends it with a significantly longer period than the measurement period. The module stores records from performed measurements in the "history" table, which it empties after each message is sent. In the table, each read electricity meter has four records allocated for each measurement performed, one for each read variable. For the electricity meter with index "0", records ID(1) to ID(4) are permanently reserved, for the electricity meter with index "1" records ID(5) to ID(8), etc. We can display the current content of the history table (i.e., the list of readings waiting to be sent) using the "**hdata**" command. Example:

```

cfg#hdata
Show history data :
ID[1] val 25514787, time 2021-01-01, 0:10:00+01
ID[2] val 0.152, time 2021-01-01, 0:10:00+01
ID[3] val 0.000, time 2021-01-01, 0:10:00+01
ID[4] val 0.000, time 2021-01-01, 0:10:00+01
ID[1] val 25514787, time 2021-01-01, 0:20:00+01
ID[2] val 0.188, time 2021-01-01, 0:20:00+01
ID[3] val 0.000, time 2021-01-01, 0:20:00+01
ID[4] val 0.000, time 2021-01-01, 0:20:00+01

```

From the history table listing, it is clear that since the last data transmission to the center, two measurement periods have taken place (at "0:10:00" and "0:20:00"), in which one electricity meter (ID 25514787) was read. The value of tariff T1 is 0.152 kWh and 0.188 kWh, the values of the other two variables are zero.

The following commands are used to set other common parameters and functions of the module:

tz	<i>setting the time zone (UTC + n)</i>
time	<i>display/set hh:mm:ss of real time RTC</i>
date	<i>display/set YY.MM.DD of real time RTC</i>
vbat	<i>setting the battery threshold voltage for generating an alarm (setting)</i>
mint	<i>setting the measuring interval of AD converters for voltage and temperature measurement</i>
loca	<i>setting individual module designation</i>

Given that the NB-R4-V module can send not only the current counter value but also "historical" values stored in memory, it must have the correct real-time value ("RTC") set so that the exact time of measurement can be registered for each stored value. GSM networks usually synchronize time with Coordinated Universal Time UTC automatically, when the device logs into the network and when sending a message. The following group of commands is used to check the RTC setting.

Using the "tz" command, we set the **time zone** (Time Zone) in which the remote reading system operates. The module supports **only one** time zone, which is set in hours from UTC. Example of setting the time zone to UTC+1 (Central European Time):

```

cfg#tz 1
Tz change from 0 to 1

```

In the configuration listing, the set time zone value is displayed as:

```

Timezone : 1

```

Using the "time" or "date" command, we can display the current RTC setting. By entering either of these commands without parameters, we display the current RTC value of the module. Example:

```

cfg#time
RTC time : 15:30:17 2019-01-30
system 1548858617 : 2019-01-30, 15:30:17+01
cfg#

```

We set the RTC value using the **time** and **date** commands as follows:

```

cfg#time 0x182555
RTC time : 18:25:55 2019-01-30
system 1548869155 : 2019-01-30, 18:25:55+01
cfg#date 0x190128
RTC time : 18:26:58 2019-01-28
system 1548696418 : 2019-01-28, 18:26:58+01
cfg#

```

As evident from the example, the "time" value is given in the format "0x", the "date" value is given in the format 0xYYMMDD. When the module is put into operation in the GSM network, the RTC value will be automatically set according to the GSM network data.

Using the **"vbat"** command, we can adjust the setting of the battery threshold voltage at which the module sends a "Low Battery" alarm. The factory-set threshold value is 3.1 V (3100 mV). We recommend changing this value only in justified cases, after consulting with the manufacturer. Example of checking the current setting and changing the threshold value to 3.2 V:

```
cfg#vbat
Vbat alarm for 3100 mV
cfg#vbat 3200
Vbat alarm for 3200 mV
cfg#
```

Using the **"mint"** command, we can set the interval for measuring analog values (battery voltage, processor temperature) using the AD converter. The factory-set interval value is 120 seconds. We recommend changing this value only in justified cases, after consulting with the manufacturer. Example of checking the current setting and changing the interval to 240 seconds:

```
cfg#mint
Measure interval 120 sec.
cfg#mint 240
Measure interval 240 sec.
cfg#
```

Using the **"loca"** command, we can set an individual designation for the module. Up to 30 alphanumeric characters can be entered. The entered designation will be displayed in the "Info text" field of the optical configuration form. The designation can contain any identification data (installation site code, customer code, serial number...). Example of setting an individual module designation:

```
cfg#info NB-X 123456
Change manuf info from : '' to : 'NB-X 123456'
cfg#
```

In the "Utils" section of "HELP" list there are the **"ppm"**, **"xtset"** and **"xmco"** commands, that are intended only for module initial setup and diagnostics. **It is recommended not using these commands during operation.**

3.2 Setting module parameters using an optical converter

The module is equipped with an "IRDA" infrared optical interface, which is used for configuration using the **"BT-IRDA"** converter (from optical to Bluetooth radio). For easy application of the optical converter, the module is equipped with a circular recess ("peephole") for applying the converter with a retaining magnet. The advantage of setting via optical converter is the possibility of configuration through the "peephole" in the plastic cover of the module, without the need to open the cover. This is of great importance especially in cases where the module is sealed by silicon filling (additional modification to meet IP68 protection degree conditions).

Via the "IRDA" optical interface, those parameters that are included in any configuration form of the **"SOFTLINK Configurator"** mobile application can be set from a mobile phone. The current version of the "SOFTLINK Configurator" application supports the configuration of all basic module parameters, as well as performing those basic tests that need to be carried out at the installation site.

Figure 2 shows the identification form of the NB-R4-V module (in the yellow frame), the list of available forms (in the green frame) and the administration form (in the blue frame).

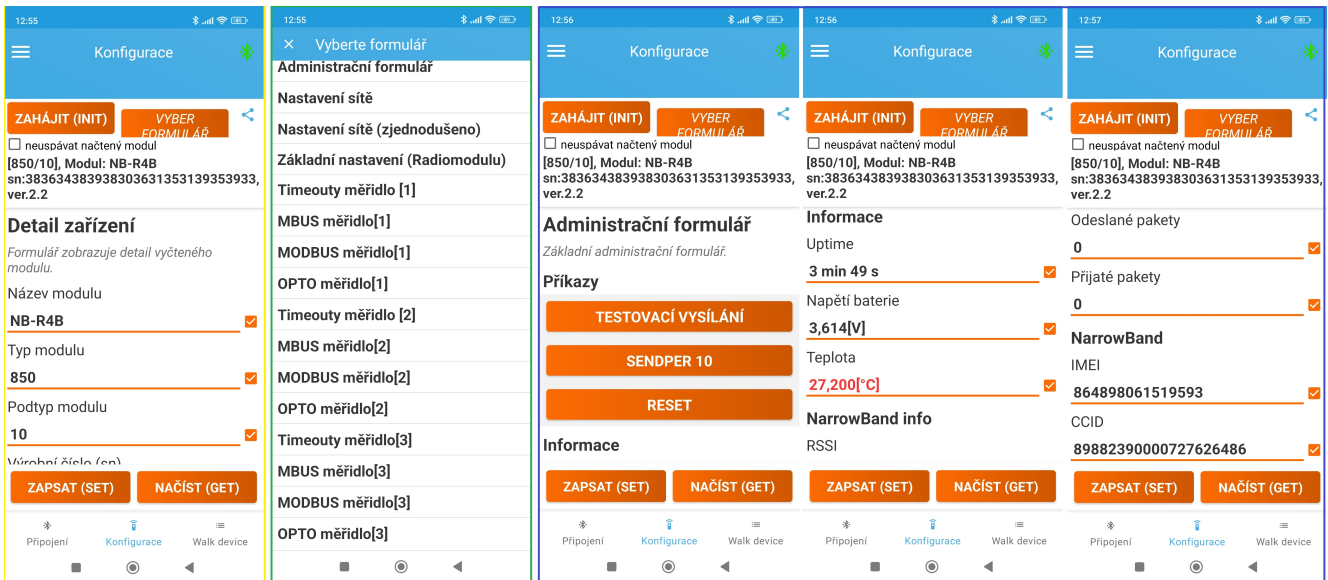


Figure 2: Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (1)

The **identification form** displays basic information about the module (type, modification, serial number, system time) and a button for selecting the configuration form. The **administration form** displays operational data of the module (uptime, battery voltage, processor temperature). There are buttons for performing a reset and turning on test transmission.

Figure 3 shows the form for setting network communication (in the purple frame), a simplified form for basic module settings (in the orange frame) and a form for setting bus timers (in the gray frame).

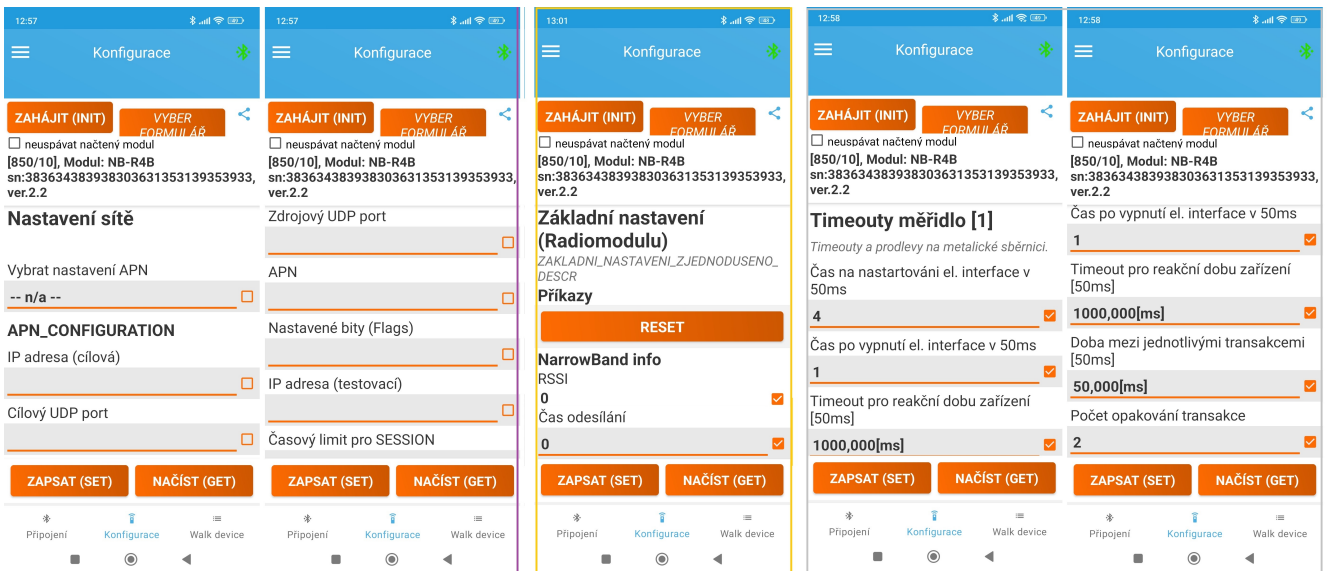


Figure 3: Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (2)

The **network configuration** form contains configuration fields for setting communication with the NB-IoT network (IP address, UDP port, APN, address for ICMP tests) and a switch for selecting the IP communication mode ("Flags"). The **Default Settings** form allows for basic functionality checks and setting of the repetition period for sending operational messages. The **Timeouts** forms are used to set bus timers for individual meters with index [1] to [6].

Figure 4 shows the form for setting communication with a meter using the M-Bus protocol (in the brown frame), the form for setting communication with a meter using the Modbus protocol (in the red frame) and the form for setting communication with a meter using the OPTO protocol (in the pink frame). All these forms are available 6 times, with index [1] to [6]. Only one protocol can be set for each index, so only the relevant form is filled in.

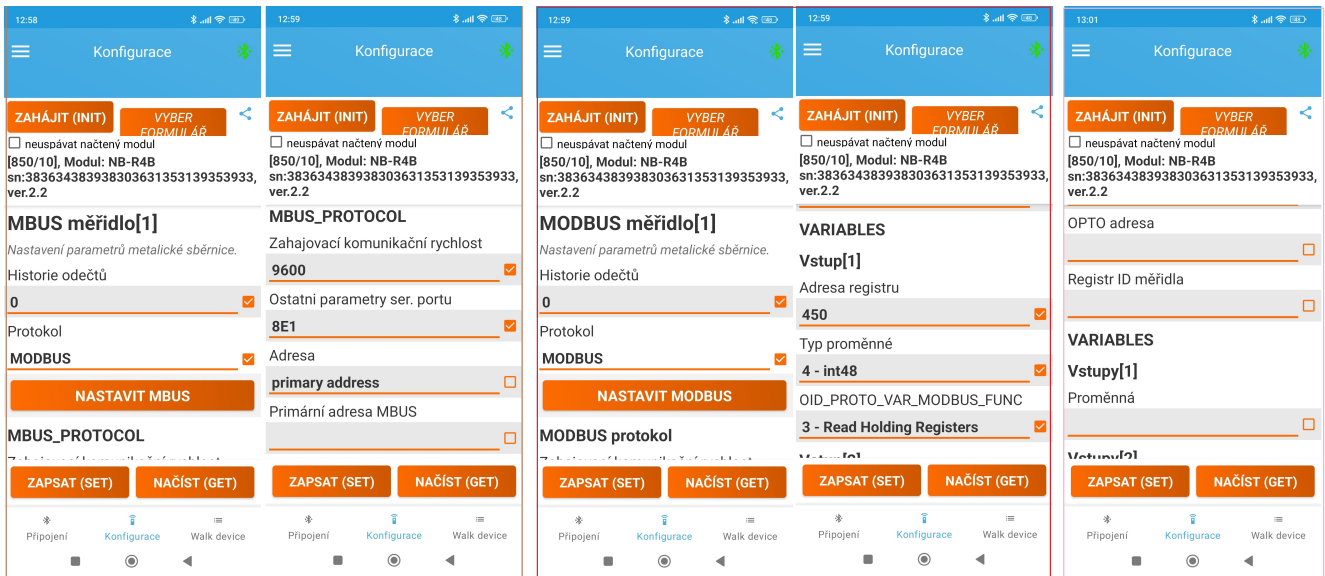


Figure 4: Forms of the NB-R4-V module in the "SOFTLINK Configurator" application (3)

Using the "**Test Transmission**" button, a message with the last read data can be sent immediately. Using the "**SENDPER 10**" button, a series of 10 messages with the last read data can be sent immediately. Both functions are used to verify module functionality during installation. Using the "**Test Interface**" button, preset registers can be immediately read from the given meter with the same effect as using the "iread" function of the electricity meter when configuring by cable. Using the "**Reset**" button, a module reset can be performed.

The "SOFTLINK Configurator" mobile application is continuously being developed and improved, so the above previews of information and configuration forms of the NB-R4-V module may change over time.

3.3 Setting module parameters from a remote computer using the reverse channel

The NB-IoT network communicates via the standard Internet Protocol (IP), which naturally enables **communication in both directions**. The NB-R4-V module uses the possibility of bidirectional communication for remote parameter setting from a remote computer via the so-called "**reverse channel**", which, to save battery capacity, opens only for two seconds after sending a message (INFO, TRAP, or RESPONSE). During this time, the module's receiver is open and the module is able to receive a message from a remote server.

Messages in the reverse direction are used to set module parameters. These "**setting messages**" are encoded by the NEP protocol, so they have essentially the same structure as messages sent by the module (individual variables in NEP encoding are transmitted in the data content of the UDP packet).

The first variable in each setting message is always the **message type**. Setting messages are always of type "**SET**" (OiD 63 = "1"). This variable is followed by one or more variables for which a change is requested.

The NB-R4-V module performs the setting of the requested parameters (update of the specified variables) and sends back a message of type "**RESPONSE**" (OiD 63 = "4"), which contains the values of the changed variables after the change is made. The module sends the RESPONSE type message either to the IP address from which the SET type request came, or to the set target server IP address (depending on the setting of the "Reply" parameter by the "sreply" command).

Using the setting messages of the reverse channel, the same parameters can be set as when setting the module using an optical converter, which communicates with the module on the same principle. More detailed information about communication possibilities via the reverse channel can be obtained by inquiring with the module manufacturer.

3.4 Data messages of the NB-R4-V module

3.4.1 Structure and types of module data messages

The NB-R4-V module is used for reading data from consumption meters via the RS-485 interface and sending current data to a superior automatic data collection system via the NB-IOT service of a GSM operator.

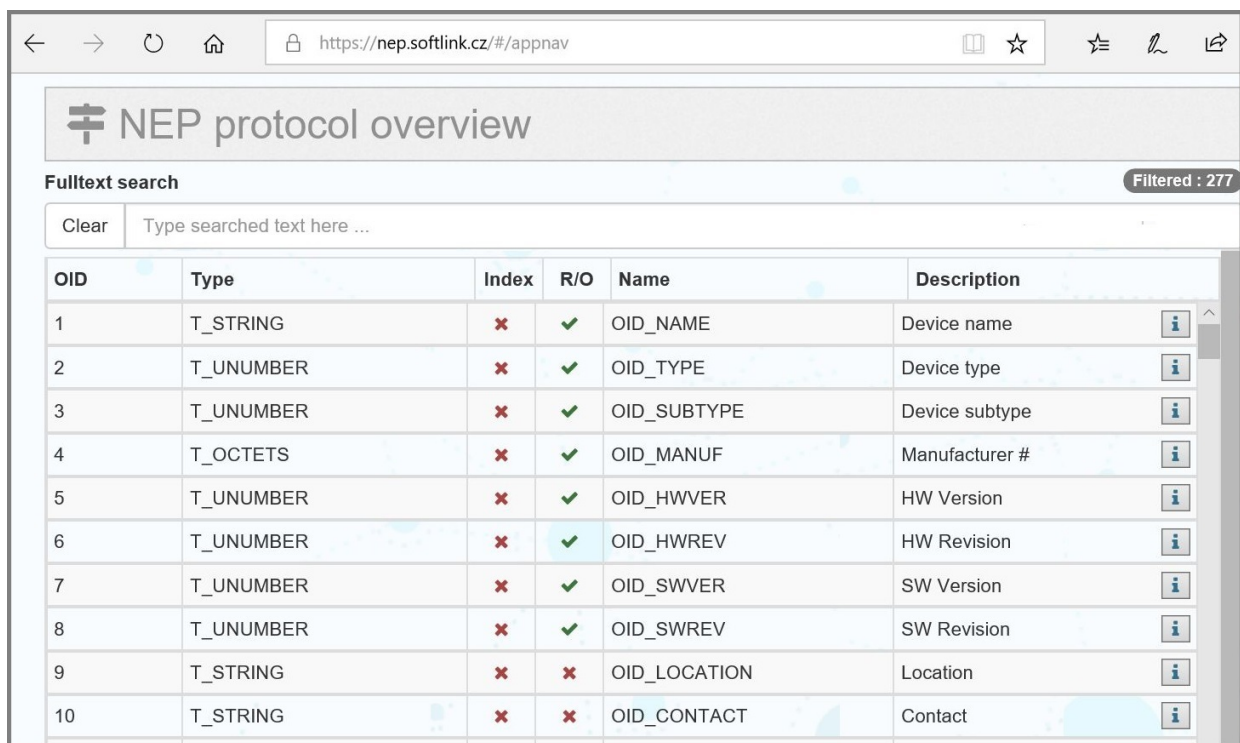
NB-IOT services use UDP (User Datagram Protocol) messages for data transfer, which is the transport layer of the Internet Protocol (IP).

The UDP datagram header of the NB-R4-V module consists of three fields:

- source port (16 bits) - fixed to "2000"
- destination port (16 bits) - set by the "Server port" parameter
- length (number of bytes) of UDP packet (16 bits)

The UDP packet header is followed by the packet data content, in which individual variables are transmitted.

Individual variables are coded into the data content of the message by using of "NEP" proprietary coding system invented by SOFTLINK. In this system each type of variable has its own designation called "OID" (Object ID), which determines meaning, character and data type of the variable. These variables, that could be used multiple times (as multiple inputs, temperatures, voltages...) must be used jointly with order number of the variable called „Index”. „NEP coding table” is centrally maintained by SOFTLINK and it is available on the public WEB address [NEP Page](#). Preview of „NEP coding table” for coding of variables in the WACO system is shown in the figure 5.



OID	Type	Index	R/O	Name	Description
1	T_STRING	✗	✓	OID_NAME	Device name
2	T_UNUMBER	✗	✓	OID_TYPE	Device type
3	T_UNUMBER	✗	✓	OID_SUBTYPE	Device subtype
4	T_OCTETS	✗	✓	OID_MANUF	Manufacturer #
5	T_UNUMBER	✗	✓	OID_HWVER	HW Version
6	T_UNUMBER	✗	✓	OID_HWREV	HW Revision
7	T_UNUMBER	✗	✓	OID_SWVER	SW Version
8	T_UNUMBER	✗	✓	OID_SWREV	SW Revision
9	T_STRING	✗	✗	OID_LOCATION	Location
10	T_STRING	✗	✗	OID_CONTACT	Contact

Figure 5: Preview of „NEP coding table” for coding of variables in WACO system

Each variable is transferred together with its decoding information „Type” and „Length” that enables decoding of the information (i.e. determine variable’s OID, index and value) on the receiving side even without knowledge of variable meaning. More detailed description of the NEP protocol can be downloaded in PDF format at the [NEP Page](#).

The data content of the message has a fixed part containing identification data and operational values of the NB-R4-V module itself and a variable part of the message, which contains measured variables. The module generates two types of messages:

- periodically generated messages of type "INFO" about the status of variables (meter readings)
- alarm messages of type "TRAP" generated by the module immediately after detecting a given event

The module generates these messages either in open or encrypted mode. In addition to these basic types of messages,

the module can also generate confirmation messages of type **"RESPONSE"**, which respond to setting messages from a remote server (see paragraph 3.3).

3.4.2 Description of INFO type message

The main part of INFO type messages are readings from connected meters, scanned by the module. Along with the readings, identification and operational data of the module are also sent. The module sends INFO messages from each meter either in "online" mode or in "history" mode, with the following rules:

- if the meter is set to "online" mode (i.e., it has its own transmission period set), the module sends INFO messages immediately after reading the data with the transmission period of the given meter (see paragraph 3.1.6 "Commands for setting communication with consumption meters and sensors"). The message always contains only data from the given meter;
- if the meter is set to "history" mode (i.e., it has a period set for storing in the history table), data from this meter is sent with the main transmission period of the module (see "periode" command with index "6"). The message always contains data from all meters that are in "history" mode;
- with combined settings (some meters are in "online" mode, some in "history" mode), the module sends messages with several transmission periods simultaneously. Example:
 - meter with index "0" is in "online" mode and has a period set to "60"
 - meter with index "1" is in "online" mode and has a period set to "120"
 - meter with index "2" is in "history" mode and has a period set to "60"
 - meter with index "3" is in "history" mode and has a period set to "30"
 - the module has the main transmission period set to "360" minutes

With these settings, the module will send data from meter "0" every hour, data from meter "1" every two hours, and a joint message with data from meters "2" and "3" every 6 hours. In the joint message, meter "2" will have six sets of historical readings, meter "3" will have 12 sets of readings in the joint message.

The fixed part of the message consists of the first nine variables, which are part of every message. In the examples of messages below, the fixed data are always marked with yellow color in the OID column.

The variable part of the message in "online" mode always contains only 4 read variables (without time data). The variable part of the message in "history" mode contains, in addition to the read variables, also **time data** ("Timestamps"), which determine the times of readings of individual variables. The timestamp is always valid for all data that follow it, up to the next timestamp (or to the end of the message).

Variables are assigned to individual meters using the index "x" permanently, as follows: - for meter with index "0", indexes 1 to 4 are allocated

- for meter with index "1", indexes 5 to 8 are allocated
- for meter with index "2", indexes 9 to 12 are allocated
- for meter with index "3", indexes 13 to 16 are allocated
- for meter with index "4", indexes 17 to 20 are allocated
- for meter with index "5", indexes 21 to 24 are allocated

Example of an INFO type message with **current data** from a meter with index "0" in "online" mode:

OID	Index	OID Name	Description	Example
63		Message type	DATA/INFO type message	6
2		Device Type	Device type	850
3		Device Subtype	Device modification	12
4		Manufacturer No.	Device identification	IMEI
12		Uptime	Time since last reset (sec)	186552
61		Sequence No	Unique message number	
105	1	Temperature	Processor temperature in tenths of a degree Celsius	223
106	1	Voltage	Battery voltage in mV	3765
462	1	RSSI	Last RSSI value	-61
100	1	Input value 1	Current value of variable var1	45628533
100	2	Input value 2	Current value of variable var2	12447
100	3	Input value 3	Current value of variable var3	0
100	4	Input value 4	Current value of variable var4	0

Example of an INFO type message with **historical data** from two meters with indexes "2" and "4" in "history" mode:

OID	Index	OID Name	Description	Example
63		Message type	DATA/INFO type message	6
2		Device Type	Device type	850
3		Device Subtype	Device modification	12
4		Manufacturer No.	Device identification	IMEI
12		Uptime	Time since last reset (sec)	186552
61		Sequence No	Unique message number	
105	1	Temperature	Processor temperature in tenths of a degree Celsius	223
106	1	Voltage	Battery voltage in mV	3765
462	1	RSSI	Last RSSI value	-61
<i>First TimeStamp and data valid for this time</i>				
17		Timestamp	Reading time (Epoch Unix Time Stamp)	1549031954
100	9	Input value 9	Current value of variable var1	44832254
100	10	Input value 10	Current value of variable var2	3257.2
100	11	Input value 11	Current value of variable var3	2159.3
100	12	Input value 12	Current value of variable var4	0
100	17	Input value 17	Current value of variable var1	32654487
100	18	Input value 18	Current value of variable var2	8249
100	19	Input value 19	Current value of variable var3	0
100	20	Input value 20	Current value of variable var4	0
<i>Second TimeStamp and data valid for this time</i>				
17		Timestamp	Reading time (Epoch Unix Time Stamp)	1549033754
100	9	Input value 9	Current value of variable var1	44832254
100	10	Input value 10	Current value of variable var2	3259.8
100	11	Input value 11	Current value of variable var3	2159.3
100	12	Input value 12	Current value of variable var4	0
100	17	Input value 17	Current value of variable var1	32654487
100	18	Input value 18	Current value of variable var2	8267
100	19	Input value 19	Current value of variable var3	0
100	20	Input value 20	Current value of variable var4	0

3.4.3 Description of TRAP type message

TRAP type messages are used for immediate sending of information about an event detected by the NB-R4-V module. They contain information about the type of detected event (for example, "Processor temperature exceeded limit"), which can be supplemented by one or several parameters of the event (for example "Temperature" and "Temperature limit"). In this way, the message recipient receives information that the temperature has been exceeded, supplemented by the current temperature reading and the limit that was exceeded.

The type of detected event is encoded in the variable "**Alarm code**" (OID 60 - TRAP CODE), where the value of the variable determines the type of event. The current variant of the NB-R4-V type module supports the following types of events:

- OID 60 - value "0" - event type "RESET"
- OID 60 - value "1" - event type "CONFIGURATION CHANGE"
- OID 60 - value "19" - input in "LOW BATTERY" state - alarm state
- OID 60 - value "20" - input in "BATTERY OK" state - normal state

The "RESET" type event is always generated by the module after it has gone through a reset (immediately after startup).

The "CONFIGURATION CHANGE" type event is generated when a new configuration set is saved to the module's FLASH memory.

The "LOW BATTERY" type event is generated when the power battery voltage drops below the set threshold value (see the use of the "vbat" command in paragraph 3.1.11).

The fixed part of the message consists of the first six variables, which are the same as in the INFO type message. Unlike the INFO type message, however, the "Message type" variable (OID 63) is set to **value "5"**, which is the flag for a **TRAP** type message.

This part is always followed by the **"Alarm code"** variable (OID 60 - TRAP CODE), which carries information about the type of event. The **"RESET"** type event corresponds to **value "0"**.

The **"Alarm code"** variable may be followed by several other variables that specify the parameters of the event. For example, for a **"RESET"** type event, it is always one variable of the **"Reset code"** type (OID 14 - RESET CODE), which carries information about what caused the reset. In NEP coding, these types of resets are defined:

- value "0" - Cold start
- value "1" - Warm start
- value "2" - Watchdog reset
- value "3" - Error reset
- value "4" - Power reset

Example of a **"TRAP"** type message with information that the NB-R4-V module has gone through a **"Warm start"** type reset (reset given by a regular command):

OID	Index	OID Name	Description	Example
63		Message type	TRAP type message	5
2		Device Type	Device type	850
3		Device Subtype	Device modification	12
4		Manufacturer No.	Device identification	IMEI
12		Uptime	Time since last reset (sec)	0
61		Sequence No	Unique message number	
60		Trap code	RESET alarm code	0
14		Reset code	WARM START reset code	1

The **"CONFIGURATION CHANGE"** type event carries the **"configuration status"** variable (OID 15 - Configuration Status) as additional information. The **"LOW BATTERY"** type event carries the **"battery voltage"** variable (OID 106 - Voltage) as additional information.

3.4.4 Principle of message encryption

Message encryption using an AES key is turned on by setting the encryption key using the **"ekey"** command as described in paragraph 3.1.11 **"Commands of the "Utils" group for setting and controlling basic module functions"**. The message is marked as an **"Encrypted message"** in the first variable (**"Message type"**) (OID 63 has the value **127 - ENCRYPTED MESSAGE**). The first six variables of the message are always sent unencrypted because they contain identification data and auxiliary data for decryption. The other variables are encrypted using **CFB block encryption** and are transmitted in the message as a single encrypted variable **"Encrypted part of the message"** (OID 19 ENCRYPTED BLOCK).

The structure of an encrypted message always looks like this:

OID	Index	OID Name	Description	Example
63		Message type	ENCRYPTED MESSAGE type message	127
2		Device Type	Device type	850
3		Device Subtype	Device modification	12
4		Manufacturer No.	Device identification	IMEI
12		Uptime	Time since last reset (sec)	186552
61		Sequence No	Unique message number	
19		Encrypted block	Encrypted part of the message	other variables

In the encrypted part of the message, all other variables are block-encrypted. The first variable in the encrypted block is always **"Message type"** (OID 63 MESSAGE TYPE), which determines whether it is an **INFO** type message (value 6) or a **TRAP** type message (value 5). Other variables follow in the same composition and order as in an unencrypted message (starting from the seventh variable to the end of the message).

4 Operational conditions

This section of the document describes basic recommendations for transportation, storage, installation and operation of NB-R4-V radio modules.

4.1 General operational risks

The NB-R4-V radio modules are electronic devices powered by their own internal battery, which register the status of counters of connected consumption meters.

During operation of the device, the following risks are particularly present:

4.1.1 Risk of mechanical and/or electric damage

The devices are enclosed in plastic boxes, so that the electrical components are protected from the direct damage by human touch, tools, or static electricity. In normal operation no special precautions are needed, besides avoiding of the mechanical damage from strong pressure or shocks.

Special attention is required for cables that connect the radio modules with the meters, sensors, or external antennas. In operation it is necessary to ensure that the cables are not stressed by mechanical tension or bending. In case of damage of any cable isolation it is recommended to replace the cable immediately. If the module is equipped with a remote antenna on a coaxial cable, much attention should be paid for the antenna and the antenna cable as well. The minimum bending radius of the antenna cable with 6 mm diameter is 4 cm, for the antenna cable with the 2,5 mm diameter the bending radius is 2 cm. Violation of these bending parameters can lead to breach of homogeneity of the coaxial cable that can cause reducing of radio range of the device. Further it is necessary to ensure that the connected antenna cable will not stress the antenna connector of the device by tension or twist. Excessive loads can damage or destroy antenna connectors.

Installation of the module can be performed only by a person with necessary qualification in electrical engineering and at the same time trained for this device installation. It is recommended to lead antenna and signal cables as far from 230/50 Hz power cables as possible.

4.1.2 Risk of premature battery discharge

The devices are equipped with the long duration batteries. Battery life can be influenced by these factors:

- storage and operation temperature – in high temperatures the spontaneous discharging current increases, in low temperature the battery capacity reduces;
- frequency of radio-transmitting.

Modules are delivered with preset period of regular transmitting of info-messages as stated in the configuration table in section of this document and the battery life cycle is quoted for this period. If the transmitting period is significantly reduced, battery life will be proportionally shortened.

4.1.3 Risk of damage by excessive humidity

Radio modules could be (as any other electronic devices) damaged by water, that could cause a short-circuit among some electronic elements or corrosion of the elements. Correctly assembled plastic box protects the module's printed circuit board against direct penetration of water, but the damage could be caused also by gradual penetration of humid air which can cause corrosion or other damage by condensed water inside the box.

Modules are enclosed in IP65 grade plastic boxes (proof against short-time squirted water) or with additional sealing by high-adhesion silicon filling, that can ensure proof against inundation by water (IP68 grade). Modules, that are delivered with IP68 sealing from factory are clearly assigned by IP68 degree of protection on the manufacturer's production label (e.g.: "NB-R4-V/B13/IP68").

Risks of damage of the device in basic "IP65" design caused by penetration of excessive humidity can be eliminated by these precautions:

- install only modules that are correctly assembled, with undamaged box and undamaged rubber seal;
- in case of any doubt perform additional sealing of connection of both parts of the box and both cable bushings by silicon sealant;
- install modules only to the sites where relative humidity exceed value of 95% only occasionally;

- install modules only to the sites where they can be squirted or sprayed by water only occasionally and only for a short time;
- do not install modules to the sites where they can be dipped into the water.

Risks of damage of the device in waterproof "IP68" design caused by penetration of excessive humidity can be eliminated by these precautions:

- do not open the module with silicon filling without serious reason;
- if (from some reason) the module was already opened, manipulate with it very carefully or renew its silicon filling by pouring of a few milliliters of special silicon (same as original - consult the technique with manufacturer). **In case the module has been opened, there is no manufacturer's guarantee of IP68 degree of protection.;**
- install modules only to the sites where they can be dipped into the water only occasionally and only for a short time;
- do not install modules to the sites where their antenna could be submerged under water. Antenna must be installed to such place, where there is no possibility to be flooded. **Operating of the module with antenna submerged under water could cause irretrievable damage of the device!**

4.2 The condition of modules on delivery

Modules are delivered in standard cardboard boxes. The modules are commonly delivered with battery switched off. There is an exception in case the modules are delivered with additional sealing by silicon filling - in this case the modules are switched on.

4.3 Modules storage

It is strongly recommended to store the modules in dry rooms or halls, in the temperature interval (0 ÷ 30) °C. To prevent the unwanted discharging of internal battery it is recommended storing the modules with batteries disconnected and activate the battery during mounting (with exception of modules with additional sealing by silicon filling - see paragraph 4.2).

4.4 Safety precautions

Warning! Mechanical and electrical installation of the NB-R4-V module can be provided only by a person with necessary qualification in electrical engineering.

4.5 Environmental protection and recycling

The equipment contains non-rechargeable lithium battery. It is necessary to remove battery before module disposal and dispose battery separately in compliance with the dangerous waste disposal rules. Damaged, destroyed or discarded devices cannot be disposed as household waste. Equipment must be disposed of in the waste collection yards, which dispose electronic waste. Information about the nearest collection yard can be provided by the relevant local (municipal) authority.

4.6 Module installation

The NB-R4-V radio modules are enclosed in plastic boxes with IP65 or IP68 protection, prepared for wall or pipe mounting. The battery switch, configuration connector, antenna connector and bus connection terminal block are located on the printed circuit board, so access to them is only possible after opening the box.

Modules with additional silicon filling sealing (IP68 protection degree) have antennas connected during manufacturing and are delivered with power on. **It is strongly recommended opening these modules during operation only when absolutely necessary and proceeding with utmost caution.** It is recommended performing installation, replacement, or configuration of these modules strictly using the BT-IRDA optical converter as described in section 3.2 "Setting module parameters using optical converter".

Figure 6 shows the NB-R4-V module disassembled into individual components.



Figure 6: Assembly of the NB-R4-V module with a rod antenna

Figure 7 shows a detail of the module's printed circuit board, highlighting the location of the configuration connector (outlined in red), the NB-IoT radio antenna connector (marked in blue), the RS-485 bus connection terminal block (marked in purple), the battery switch (marked in yellow) and the SIM card holder (marked in green). The serial number on the module label must always match the serial number on the auxiliary label attached to the printed circuit board (marked in orange). The appearance of the printed circuit board may vary slightly depending on the module modification.

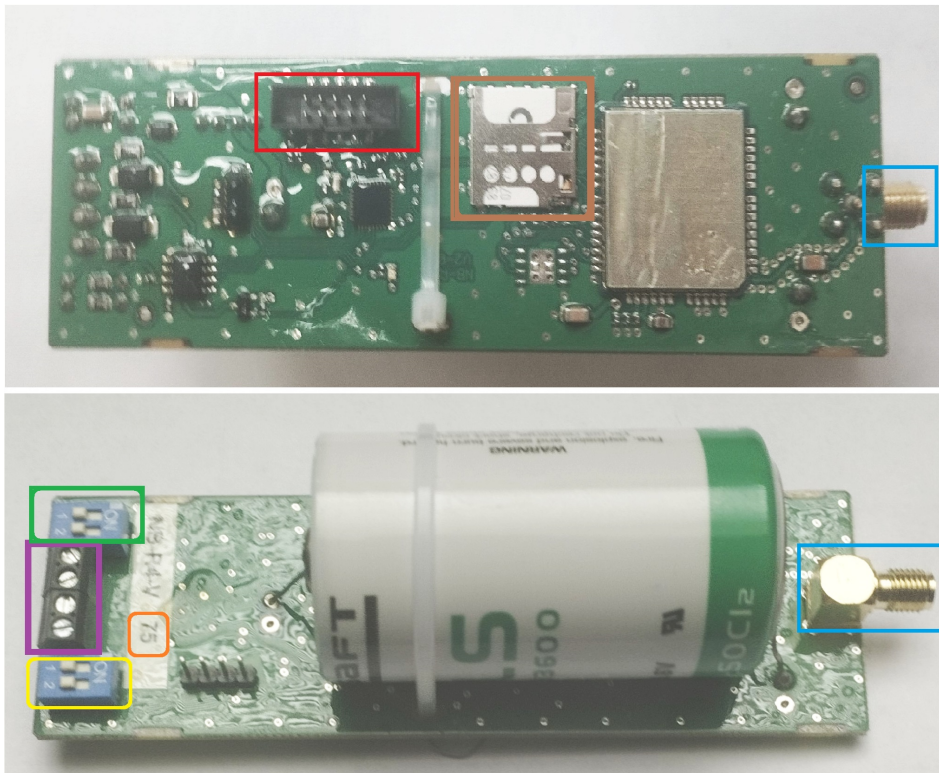


Figure 7: Detail of the printed circuit board of the NB-R4-V module

Figure 8 shows a detail of connecting the read device to the NB-R4-V module terminal block via the RS-485 bus interface.



Figure 8: Connection of a meter with RS-485 output to the NB-R4-V module terminal block

The case consists of two parts:

- the module housing into which the printed circuit board is inserted. On this part of the box there is a label, a window for magnetic attachment of the USB-IRDA/BT-IRDA converter, a cable grommet and moldings for attaching the module;
- box lid, closing the housing. There is a second cable grommet on the lid.

Installation of a module that already has the NB-IoT antenna connected, is pre-configured, turned on and has the bus cable already led out, is performed as follows:

- attach the module to a suitable solid object (wall, pipe...) using four screws or a cable tie. Use the mouldings on the bottom of the module housing for mounting. The recommended mounting position is vertical, with the lid at the bottom;
- connect the cable from the meter, or (in parallel) cables from multiple meters, to the auxiliary extension/distribution terminal block outside the module;
- using the USB-IRDA/BT-IRDA converter and the "SOFTLINK Configurator" mobile application, check the module configuration and use the "Read" button to read all connected meters;
- check that the cap nuts on both cable grommets are tightened to seal both grommets;
- if the installation procedure or the customer's internal rules require sealing of the module (as protection against possible tampering), seal the module in the specified manner (for example, by sticking a seal across the joint between the two parts of the case).

Before installing a module that is not yet assembled, or is not turned on, or needs to be set up using a cable, we must first open the module, assemble it, turn it on and configure it. These operations are performed in the following steps:

- completely loosen the cap nuts of the cable grommets at both ends of the module;

- unscrew the two screws on the sides of the box to release and remove the module lid;
- carefully slide the printed circuit board (PCB) out of the module housing. Either slide the board out completely (if it's necessary to screw on the NB-IoT antenna), or only partially so that the configuration connector is outside the housing (see figure 7). If the NB-IoT modem antenna is already installed, help yourself by gently pushing the antenna into the module when sliding out the PCB;
- if the NB-IoT antenna has not been mounted on the printed circuit board, screw it to the antenna connector at the end of the module;
- loosen the screws on the terminal block for connecting the bus cable, feed the bus cable through the grommet in the module lid and connect the bus cable wires (*) to the corresponding terminals of the terminal block according to the label on the top side of the box lid (see figure 9);
- switch the micro-switch ("jumper") located on the printed circuit board to the "ON" position to connect power to the module;
- perform basic diagnostics of the module and possibly its configuration (parameter setting) using a cable according to the procedure described in section 3 "Configuration of module parameters". If the module was pre-configured in the preparatory phase of installation, it is recommended performing at least a test reading of all connected meters using the "Read" button in the mobile application;
- insert the printed circuit board into the module housing. Insert the board so that the battery micro-switch is on the open side of the housing (i.e. on the side where the lid will be screwed on). The cap nut of the housing cable gland must be completely loosened so that the antenna (or antenna cable) can easily slide out through the gland from the housing. Push the board all the way in by pressing with your finger on the edge of the PCB (don't push on the terminal block or micro-switch). In the correct position, the printed circuit board should protrude from the edge of the box housing by only about 7 mm.
- check the integrity of the rubber seal on the edge of the housing and make sure that the cap nut on the lid is completely loosened and the cable to the optical head(s) moves freely through it;
- carefully slide the lid onto the box housing. The bus cable gradually slides out through the lid gland. Attach the lid to the housing by screwing in and tightening both screws;
- tighten the cap nuts on both cable grommets to seal both grommets;
- if the installation procedure or the customer's internal rules require sealing of the module (as protection against possible tampering), seal the module in the specified manner (for example, by sticking an adhesive seal across the joint between the two parts of the box).

(*) *If there are multiple meters connected to the module, we recommend connecting only one cable to the input terminal block, which will bring the bus to a suitable space near the meters (for example, to a local distribution box). At the end of this cable, connect a more robust auxiliary distribution terminal block suitable for connecting multiple cables. Connect the cables from individual meters to the distribution terminal block in parallel. This solution is necessary **for modules with silicone filling sealing**, because the cable to the input terminal block is connected to the module before it is filled with silicone material. In this case **do not disassemble** the module, just attach it to the installation site, connect the meters to the auxiliary terminal block and check functionality using an optical converter.*

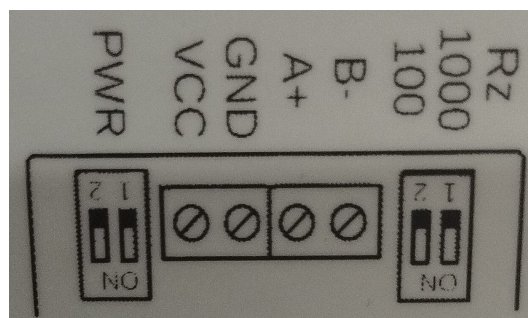


Figure 9: Diagram of connecting RS-485 bus wires to the NB-R4-V module terminal block

In general, the module has a declared degree of moisture resistance (IP65 or IP68) only if it is properly assembled and sealed. Waterproof modules with IP68 protection must be professionally sealed with silicone filling. When installing modules with IP65 moisture resistance, it is necessary to observe these principles::

- that the cable glands are properly sealed;
- that the connection point of both parts of the case is sealed with an undamaged rubber seal (included in the

delivery).

After installation record the status of the read meters in the installation protocol and possibly verify once again the functionality of the module and the correctness of the module's output values (whether they correspond to the data on the meter counters), preferably using the "end-to-end" method, i.e. by checking the display of consumption data and operational parameters of the module directly in the remote reading system.

When selecting the installation location of the module, the type and location of the antenna and the length of the antenna cable, it is necessary to take into account both the protection of the module from possible mechanical damage (installation outside operationally exposed places) and especially the conditions for radio signal propagation at the installation site. These conditions can either be determined (estimated) empirically, based on previous experience, or by measuring the signal strength using a control transmitter/receiver.

4.7 Replacement of the module and replacement of the read meter

When replacing the module due to a malfunction or due to battery depletion, proceed as follows:

- if the module was sealed, check if the seal is intact before dismantling the module. Handle a broken seal according to the internal rules applicable to the given customer/project;
- loosen the mounting screws (or tightening strap) that hold the module on the wall, pipe, or other surface and remove the module;
- if replacing the entire set (module with integrated NB-IoT antenna and bus cable led out of the module to an auxiliary terminal block), just disconnect the module from the auxiliary/distribution terminal block, replace it "piece for piece" and connect the new module to the auxiliary distribution terminal block;
- visibly mark the original module as "defective", possibly fill out the appropriate form (installation sheet) or other prescribed documentation for module replacement;
- perform a functionality check on the new module according to the procedure described in section 4.6. Pay particular attention to correctly setting the configuration parameters, especially the transmission period and communication settings with the meters;
- write down the serial number and seal number of the new module and possibly also the status of the counters of the read meters;
- if possible, immediately ensure that the new serial number is entered into the collection system database.

If not replacing the entire set, proceed with the replacement as follows:

- loosen the cap nut of the the cable grommets on the lid side;
- unscrew the two screws on the sides of the case to release the module lid and carefully slide the lid off the module. The cable (or multiple cables) to the meters slides into the lid;
- switch the micro-switch ("jumper") located on the printed circuit board to the "Off" position to turn off the module;
- disconnect the bus cable(s) from the module terminal block;
- if the module is equipped with an external NB-IoT antenna, loosen the cap nut on the module housing and carefully slide the printed circuit board out of the housing so that we have access to the antenna connector;
- disconnect the antenna cable from the antenna connector;
- reassemble the original module by screwing the lid to the housing (*). Visibly mark the module as "defective", possibly fill out the appropriate form (installation sheet) or other prescribed documentation for module replacement;
- attach the new module in place of the original one and proceed further according to the procedure described in section 4.6. Pay particular attention to correctly setting the configuration parameters, especially the transmission period and communication settings with the meters;
- write down the serial number and seal number of the new module and possibly also the status of the counters of the read meters;
- if possible, immediately ensure that the new serial number is entered into the collection system database.

(* **CAUTION!** When assembling the module, always make sure that the box housing is not mixed up, i.e. that the module PCB is completed with the original box. The serial number stated on the module housing must always correspond to the serial number on the auxiliary label that is stuck on the printed circuit board.

When replacing a meter read by the NB-R4-V module, where the reason for replacement is a meter failure, expired verification period, or other reason on the meter side, proceed as follows:

- if the meter is connected to an external auxiliary distribution terminal block, do not open the module (if possible), and set up communication with the new meter using the USB-IRDA/BT-IRDA converter and mobile application;
- using the "Read" button in the mobile application, check whether the new meter responds to queries and whether the read values match the data on its display or counters;
- if wireless configuration is not possible, check if the adhesive seal is intact and open the module according to the procedure described in section 4.6;
- connect to the module with a configuration cable and set the communication parameters with the new meter (see paragraph 3.1.6 "Commands for setting communication with consumption meters and sensors").
- using the "iread [index]" command (see paragraph 3.1.6), check whether the new meter responds to queries and whether the read values match the data on the meter's display or counters;
- fill out the prescribed documentation for meter replacement (installation sheet), especially carefully write down the status of the new meter counters;
- cover and seal the module according to the procedure described in section 4.6, or wait for the first reading to be performed;
- If possible, immediately ensure the replacement of the meter identification data in the collection system.

4.8 Dismantling the module

When dismantling, remove the module from the wall (pipe, other surface..), open it, turn off the battery, disconnect the bus cable and possibly disconnect the antenna cable. Reassemble the module by putting the lid back on the housing, properly mark it as dismantled and fill out the appropriate documentation prescribed for this case by internal regulations. If possible, immediately ensure deactivation of the module in the collection system.

4.9 Module functionality check

After putting the module into operation (or after each repair and module replacement), it is recommended checking its basic functions:

- check the setting of basic module parameters, especially the message sending system parameters (encryption, transmission period, path to the superior server) according to paragraph 3.1.10;
- check communication with meters using the "iread" and "send" commands via the configuration cable, or via the mobile application using the "Test transmission" and "Read" buttons.
- verify sufficient coverage of the installation site with NB-IoT radio signal by sending several test messages using the "send" command according to paragraph 3.1.5 "Commands of the "System commands" group for checking basic module functions" and their successful reception in the central system. Informative data on network signal availability can be obtained by checking the RSSI value in the configuration parameter listing or in the optical configuration form (value "Last RSSI");
- comprehensive (end-to-end) check of remote reading functionality can be performed by checking in the reading system whether data from all connected meters is correctly loaded. If the reading period is long, or it is not possible to wait for sending a message at the standard interval, use the immediate message sending function as described in the previous paragraph.

4.10 Operation of the NB-R4-V module

The NB-R4-V module performs remote reading of meter statuses and sending of radio messages with readings completely automatically. The greatest risks of permanent failure of radio module transmission are associated with the activities of the object user, especially the risk of mechanical damage to the modules when handling objects at the installation site, damage to the module due to water ingress, or the risk of signal shading by a metal object. A typical consequence of damage is complete loss of connection with the module.

To eliminate these risks, it is recommended paying great attention to the selection of the module installation site and the selection of the type and location of antenna installation so that a suitable compromise is found between the quality of radio connection via the NB-IoT network and the degree of risk of mechanical damage to the module, cable between the module and meter, antenna cable, or antenna. The installation itself needs to be carried out carefully, using quality cables and mounting elements.

Unexpected interruption of connection with the module can be prevented by continuous monitoring of the regularity and correctness of the read data from meters (including accompanying data on processor temperature and battery

voltage) and in case of detection of failures or non-standard values, contact the object user or perform a physical check at the installation site.

The risk of premature battery discharge can be easily eliminated by respecting the recommendations given in paragraph 4.1.2.

5 Troubleshooting

5.1 Possible causes of system failures

During operation of the NB-R4-V device, failures, malfunctions, or other operational problems may occur, which can be divided into the following categories according to their cause:

5.1.1 Power supply failures

The module is powered by an internal battery with a long lifetime. The approximate battery lifetime is specified in more detail in paragraph 1.3 "Module features". The battery lifetime is influenced by circumstances described in detail in paragraph 4.1.2 "Risk of premature internal battery discharge". Low voltage of the power supply battery initially manifests as irregular data reception failures from the given module, later the radio connection with the module is interrupted completely. The battery is soldered onto the printed circuit board and its replacement requires disassembly of the module. Battery replacement can only be performed by a person with appropriate qualifications and experience; soldering the battery by an unqualified person risks damaging the module's printed circuit board. The "NB" series modules use only the highest quality batteries that have been carefully selected and tested for this purpose. In case of battery replacement by the device user, the new battery must match the original battery as closely as possible in its parameters (type, capacity, voltage, current load, self-discharge current...). The module manufacturer strongly recommends using the same type of battery for replacement as was used in the module during its production.

5.1.2 System failures

System failures are considered to be mainly processor failures, memory failures, internal power supply failures, or other fatal failures that cause complete device malfunction. If the device is in a state where the battery has the correct voltage and shows no signs of discharge, yet the device does not communicate through the configuration port, does not respond to any configuration commands, and this state does not change even after performing a module restart, it is likely a system failure. We perform device replacement according to paragraph 4.7 and then perform setup and functionality check of the new (replaced) device. If the new device functions normally, we mark the original module as defective and record the replacement data in the operational documentation according to internal rules.

5.1.3 NB-IoT network communication failures

Functionality of transmission to the NB IoT network is indicated by the flashing of a yellow LED on the printed circuit board. If the module's power supply has the correct voltage, the module communicates through the configuration port, responds to configuration commands, and yet no messages are received from it, the cause may be a fault related to radio signal transmission or reception. A typical symptom of transmission and reception faults are also states of "partial" functionality, which manifest especially in frequent dropouts in data reception from the module. The cause of the above-described communication faults of the module may be unreliable radio data transmission, which may be caused by:

- weak NB-IoT network radio signal at the installation site. Network signal availability may change over time depending on weather conditions (fog, rain...), or as a result of changes at the transmission site and its surroundings (for example, change in the location of the base station antenna by the network operator, or construction activity in the vicinity of the base station);
- permanent or temporary signal shading due to construction modifications in the building of the module installation site, or due to operations in the given building (movement of mechanisms, machines, cars near the device);
- permanent, periodic, or irregular radio interference of the radio network by parasitic signal from an external source (operation of another system in the same radio band, industrial interference);
- low level of transmission signal, caused by a fault in the module's transmitter;

- low level of received signal due to a fault in the module’s receiver;
- damage to the antenna or antenna cable (only for module types with external antenna).

If the above-described symptoms of unreliable radio transmission occur, we proceed as follows when searching for and eliminating the causes of the problem:

- we perform a visual inspection of the module installation site and determine whether there have been any construction modifications or other changes in the building that could affect the propagation of the radio signal. We address any negative impacts of such changes and modifications organizationally, or (if possible) by changing the location of the device, or by relocating the antenna (for modules with external antenna);
- for modules with external antenna, we perform a visual inspection of the antenna and antenna cable, possibly also replacing these components with other components with verified functionality;
- we check the configuration parameters of the module and check the functionality of the module according to paragraph 4.9;
- we replace the module according to paragraph 4.7 and then set up and check the functionality of the new (replaced) module according to paragraph 4.9;
- if after performing the replacement under the circumstances described in the previous point, the replaced module also does not work correctly, the cause of the problem may be local radio interference, or the cause is insufficient network signal at the installation site. In this case, we consult the current status and possible future development of NB-IoT network signal coverage at the installation site with the service provider.

5.1.4 Communication faults with meters and sensors

Data bus faults manifest as complete or partial malfunction of communication over the bus. A module with a malfunctioning data bus communicates through the configuration port, responds to configuration commands, but messages from all or some devices (meters, sensors) on ”its” bus do not pass through to the radio network. In some cases, there may be partial malfunctions of communication over the bus, where either time-limited outages occur, or communication over the bus only fails with some devices (meters, sensors).

Faults and outages of communication over the data bus can be caused by these reasons:

- incorrect setting of communication speed and other parameters for communication with the given device over the bus;
- mechanical damage to the bus cable;
- failure of the module’s line amplifier;
- reduction of transmission properties of the bus due to changes and modifications to the bus (adding another device, changing the order, cable replacement, connecting or disconnecting the terminating resistor...);
- interference with the modulation of the electrical signal in the bus by induction of interfering signal into the bus cable, or problems caused by high potential difference of devices on the bus.

Recommendation: *General problems with the transmission properties of the bus, described in the last two points, mainly manifest in buses with a large total length and with a high number of connected devices. It is recommended entrusting the search for causes and elimination of faults of this type to an expert with appropriate knowledge who has experience with the operation of the given type of bus.*

If there is a suspicion that a possible operational problem with data collection from devices on a remote data bus may be caused by a communication fault over the data bus, first make sure that the data collection is set correctly at the logical and application level, especially whether the identification (addressing) of individual elements in the central data collection system is correctly set. If the correctness of the identification setting of individual devices is confirmed, proceed with locating and eliminating the cause of the problem with the functionality of the data connection with the connected meter/sensor as follows:

- visually check the correctness of the connection of the bus cable from the given meter/sensor to the module and check the integrity of the cable using an ohmmeter. If the cable shows signs of damage or is non-functional, repair or replace it;
- if the bus cable is undamaged and messages from other devices on the bus arrive to the central system properly, check the consistency of the bus communication parameter settings for the given device with the specification and parameter settings of the given device (see description of bus parameter settings in paragraph 3.1.6);
- if the bus is physically functional, the setting of communication parameters of the NB-R4-V module for individual devices is correct and in accordance with the settings of individual meters/sensors, but communication over the bus still does not work, the module is probably faulty and needs to be replaced according to paragraph 4.7;

The correctness of data reading from individual devices on the bus can be verified using the ”iread” command (see paragraph 3.1.6 ”Commands for setting the bus”).

5.2 Procedure for determining the cause of failure

When identifying the probable cause of a failure, proceed as follows:

1. If no data is being read from any meter/sensor connected to the NB-R4-V module, check the functionality of individual module subsystems in this order:
 - verify the correct setting of the module in the remote reading system database;
 - check the power supply functionality according to section 5.1.1 "Power supply failures";
 - check the system functionality according to section 5.1.2 "System failures";
 - check the functionality of data transmission and reception according to section 5.1.3 "Communication failures with NB IoT network";
 - check the functionality of communication with read devices on the bus according to section 5.1.4 "Communication failures with meters and sensors".
2. If data is not being read from only some of the devices (meters or sensors) connected to the module's bus, check the functionality of individual module subsystems in this order:
 - check the functionality of the meter or sensor itself
 - verify the correct setting of the address of the given meter/sensor in the configuration of the central data collection system and the bus address of the meter/sensor in the NB-R4-V module
 - check the functionality of the bus according to section 5.1.4 "Communication failures with meters and sensors"
3. Data from some connected meter/sensor is incorrect. In this case, it is recommended checking the functionality of the given device.
4. Data from the module arrives irregularly, with periodic outages. In this case, it is recommended checking the functionality of individual module subsystems in this order:
 - check the functionality of data transmission and reception from the NB IoT network according to section 5.1.3 "Communication failures with NB IoT network";
 - check the power supply functionality according to section 5.1.1 "Power supply failures".

WARNING: The NB-R4-V module is a reliable device of relatively simple and durable construction, so there is a high probability that any failure is caused by external installation circumstances, especially mechanical damage, cable damage, moisture ingress, battery discharge, or radio interference at the installation site. With each module replacement due to failure, it is recommended verifying whether the cause of the failure was one of these circumstances and, if necessary, taking measures to eliminate it.

6 Additional information

This manual focuses on the description, parameters and configuration options of NB-R4-V type radio modules designed for operation in the NB-IoT network, which are part of the **wacoSystem** product family by SOFTLINK. Further information about the NB (NB-IoT), WS868 (Sigfox), WM868 (WACO), or WB169 (Wireless M-Bus) type series modules can be found on the manufacturer's website:

www.wacosystem.com
www.softlink.cz

If you are interested in any information related to the use of NB, WS868, WM868, WB169 series radio modules, or other SOFTLINK manufacturer's devices for telemetry and remote reading of consumption meters, you can contact the manufacturer:

SOFTLINK s.r.o., Tomkova 409, 278 01 Kralupy nad Vltavou, Czech Republic, Phone: +420 315 707 111,
e-mail: sales@softlink.cz, WEB: www.softlink.cz.