Wireless M-Bus

Implementation in TR-7xD-WMB and GW-USB-06-WMB

Firmware v2.21

User's Guide

Smarter wireless. Simply.

Contents

Introduction

This User's guide describes the Wireless M-Bus Meter device implemented in IQRF transceiver module TR-72D-WMB [1].

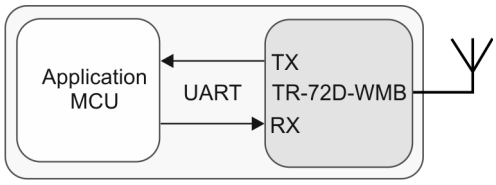
The TR-72D-WMB module works as a wM-Bus modem with the UART interface. The embedded protocol transmits and receives the wM-Bus data packets based on application data from an external application MCU. The module is configured via the UART interface using a simple service command set. Configuration parameters are stored in non-volatile memory. The TR module is automatically set in Sleep mode with very low current consumption, and woken up on a UART command.

The abbreviations used in this guide can be found in the *OMS* (Open metering System) *Specification Glossary* [3]. Hexadecimal numbers are marked with the prefix "`0x`" and binary coded numbers with "`0b`". Numbers without a prefix represent ASCII strings unless another coding is explicitly declared.

System topology

Meter

wM-Bus Meter includes the TR-72D-WMB module and an application MCU connected via the UART interface. The application MCU collects data from sensors, composes packet payload and writes/reads to/from the TR module. The TR module setup can be done before installation of the Meter using the CK-USB-04A development kit. Supported Meter communication modes are S1, T1 (unidirectional) and S2, T2 (bidirectional).

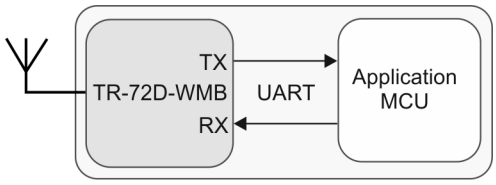


Communication with the MUC can be operated and tested in a setup depicted below (Sniffer). A SW utility for control from PC is available.

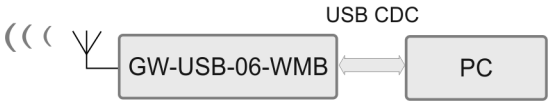
MUC

The wM-Bus MUC is a communication device collecting data from Meters. Current implementation only allows to test bidirectional communication with the Meter.

Communication with the Meter can be operated and tested in a setup depicted below (Sniffer). A SW utility for control from PC is available.



Compact MUC implemetation is possible using the GW-USB-06-WMB gateway:

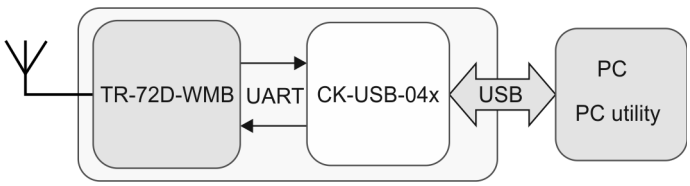


The MUC implementation in this firmware version is intended just for evaluation with the Meter in T and S modes. Complete MUC functionality will be implemented later on.

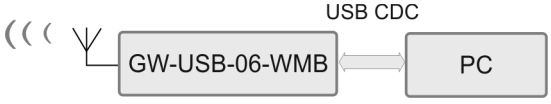
Sniffer

The wM-Bus Sniffer is implemented by TR-72D-WMB module and CK-USB-04A development kit connected via USB to PC. It allows to monitor wM-Bus communication.

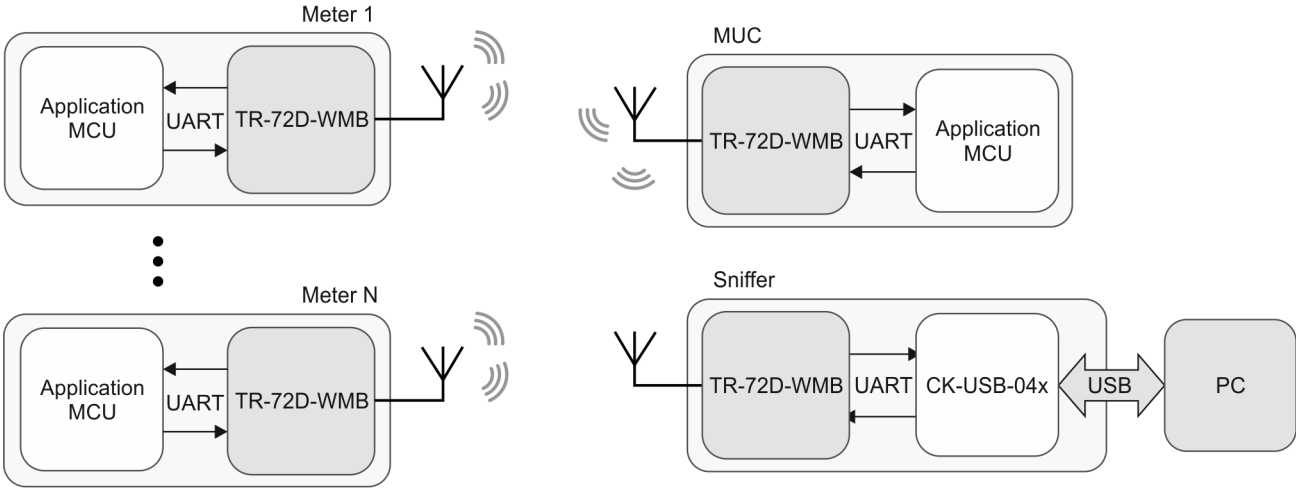
A SW utility displaying all ambient RF packets is available.



Compact Sniffer implemetation is possible using the GW-USB-06-WMB gateway:



Typical usage



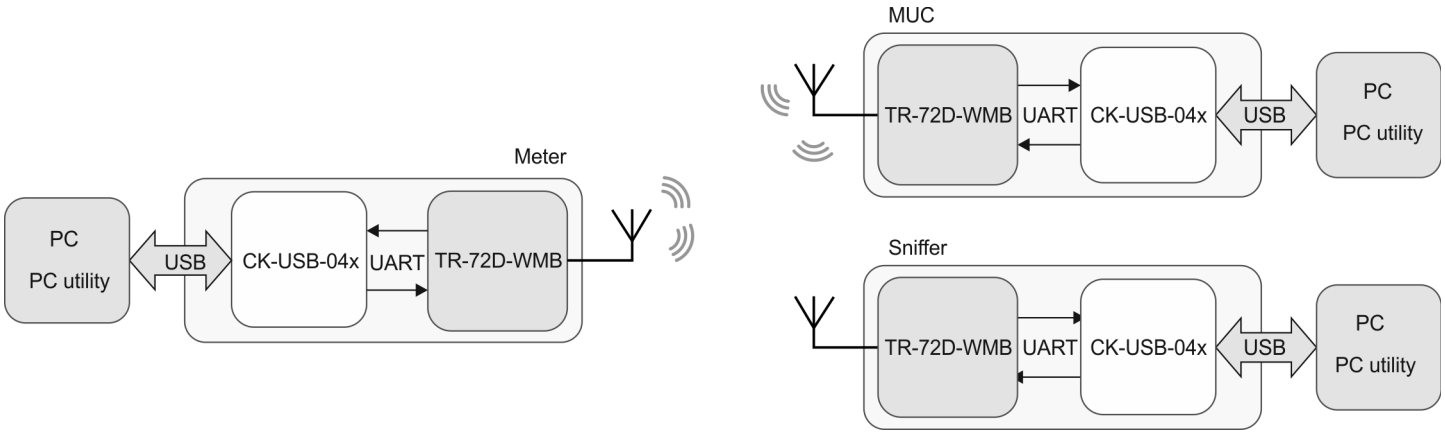
Development

To implement IQRF Wireless M-Bus, besides the TR-72D-WMB module, a control unit with UART interface embedded in a device is required. The development of wM-Bus applications can be accomplished without such specific user hardware and firmware using standard IQRF development kit CK-USB-04A with special firmware. The kit communicates with PC via USB CDC class creating virtual serial port transparently converting data UART ↔ USB.

LED indication:

- LED 1 – flashes once by data transfer from CK-USB-04A to PC
- LED 2 – flashes once by data transfer from PC to CK-USB-04A

A SW utility for control from PC is available. Then, Wireless M-Bus applications can be developed, debugged and tested as follows:

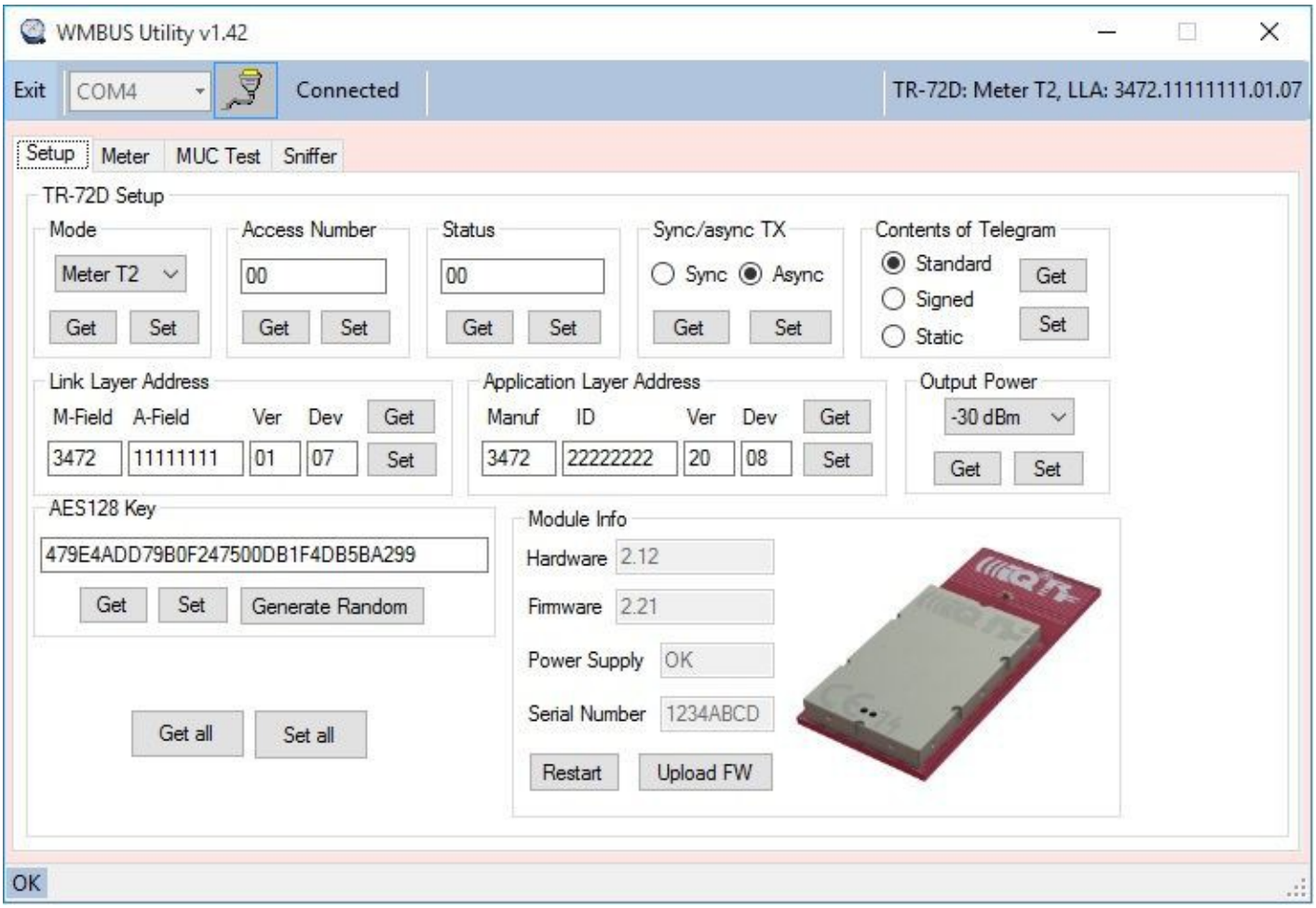


WMBUS utility

To setup parameters and control wM-Bus devices in all Meter, MUC and Sniffer modes from PC, the `WMBUS_Utility_xxxxxx-.exe` demo is provided. Proper USB CDC driver must be installed on PC. It can be downloaded from www.iqrf.org/89.

Setup

Configuration parameters of TR module can be adjusted according to requirements of given application using the *Setup* tab. The service commands listed below are used for this. Configuration parameters are stored in TR-72D-WMB internal EEPROM memory.

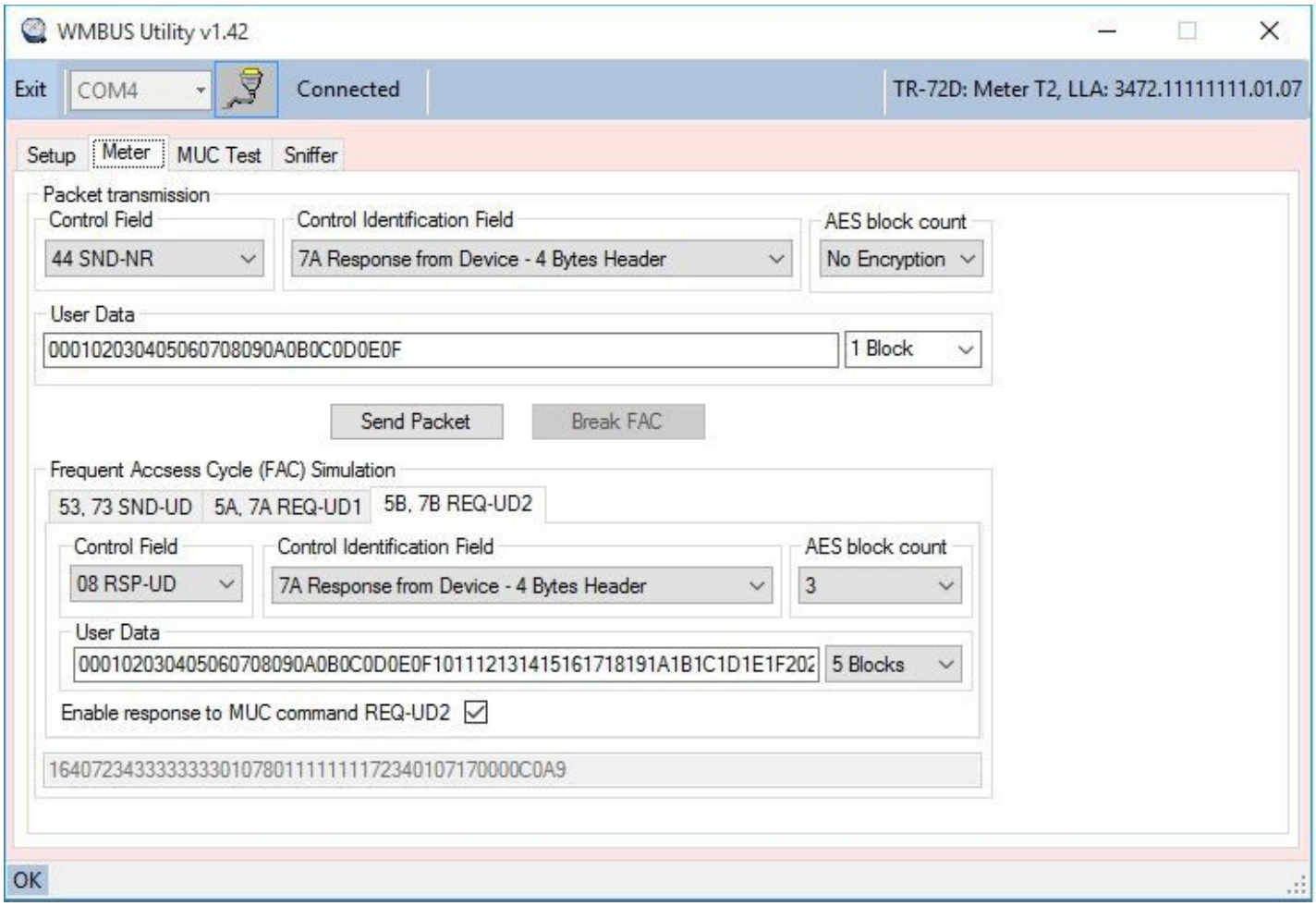


The *Upload FW* button allows user to update TR-7xD-WMB firmware. The firmware is available at http://iqrf.org/weben/index.php?sekce=support&id=upgrades as an `iqrffw` file. Follow the Upload procedure to update the firmware.

1. Download the firmware file and save it to HDD.
2. Click the *Upload FW* button. The WMBUS Utility asks you to confirm the upload operation.
3. *Select IQRF Firmware file dialog* is shown. Select downloaded `iqrffw` file and click OPEN.
4. WMBUS Utility switches TR-7xD-WMB module to bootloader mode and uploads the firmware.
5. Wait until the upload operation is completed.

wM-Bus Meter

The *wM-Bus Meter Test* tab allows to send packets according the *OMS specification* [2]. In case of bidirectional Meter (S2/T2), all packets received by the Meter (requested by the MUC) are logged and displayed in the *wM-Bus Sniffer* tab. Double-click opens a detailed view for given packet.
The bidirectional Meter (S2/T2) behavior after the Frequent Access Cycle (FAC) initiated by the MUC should be specified in the *wM-Bus Sniffer* tab (FAC simulation).



Supported packets:

- SND-NR (Send spontaneous/periodical application data without request – Send/No Reply)
- SND-IR (Send manually initiated installation data – Send installation request)
- ACC-NR (No data – Provides opportunity to access the meter between two application transmissions)
- ACC-DMD (Access demand to master in order to request new importat application data – Alerts)
- ACC (Acknowledge the reception – part of FAC Simulation)
- RSP-UD (Response of application data after a request from master – part of FAC Simulation)

wM-Bus MUC

TR-72D-WMB MUC automatically answers to the ACC-DMD packets (by the ACK packets) and to the SND-IR packets (by the CNF-IR packets) according the *OMS specification* [2]. Additionally, it enables to predefine two packets to be sent by the MUC to the Meter as a part of the FAC. This predefining is accomlished by the 0x81 command (binary mode, the same as the 0x80 command but with parameter level added).

After receiving the SND-NR or ACC-NR packets, the MUC sends the request (to the Meter) predefined by the 0x81 command `level0` (The MUC initiates FAC). If the `level0` request is missing in the 0x81 command, the MUC does not respond and no FAC is initiated.

The next received packet is an answer from the Meter to the request. Then the MUC sends a request predefined by the command 0x81 `level1`. If the `level1` request is missing in the 0x81 command, the MUC automatically answers to the Meter by the SND-NKE packets and terminates FAC.

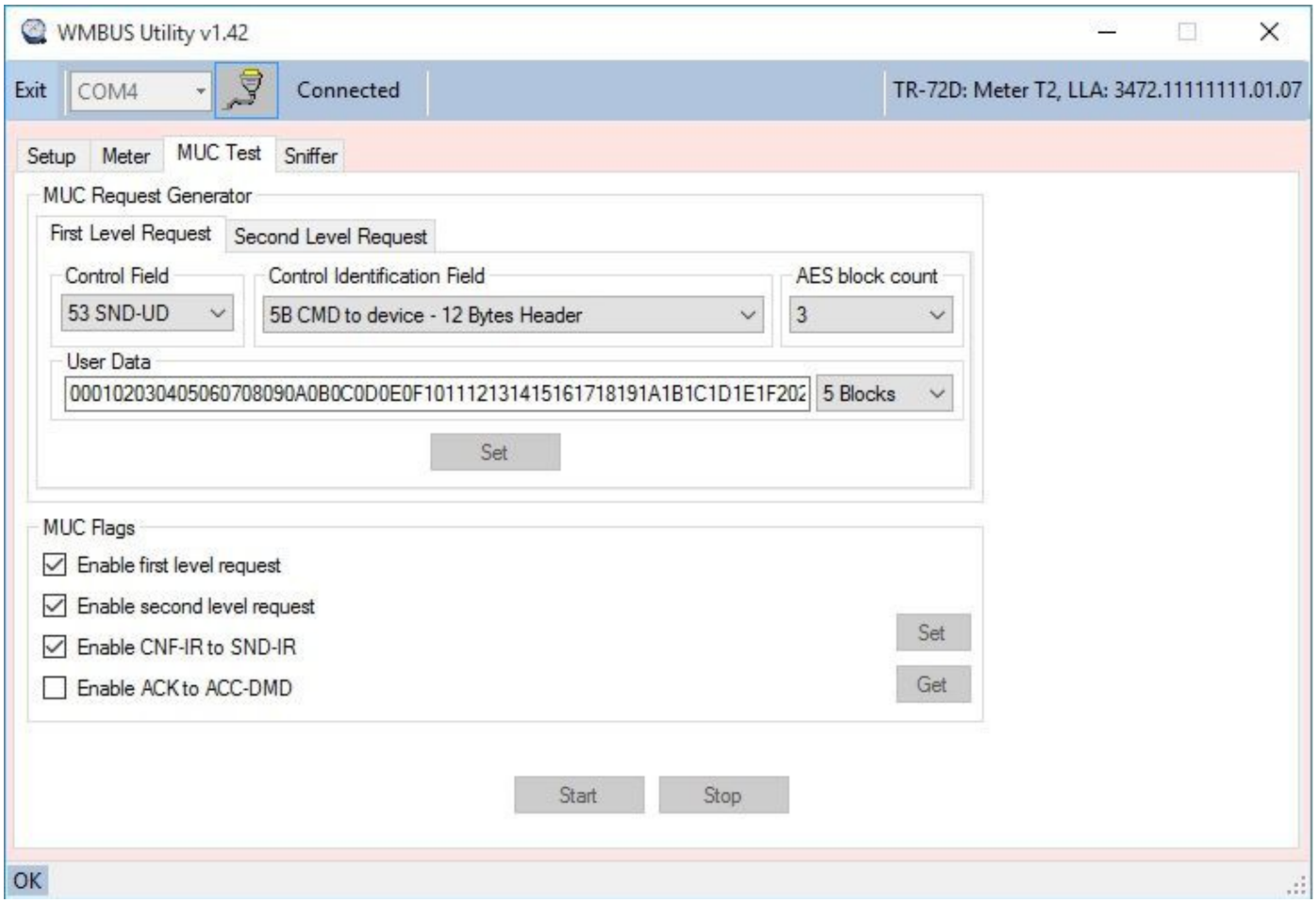
The next packet (received by the MUC as an answer from the Meter to the `level1` request) is automatically answered by the MUC by the SND-NKE packet and FAC is completed.

Packets received by the MUC are logged and displayed in the wM-Bus *Sniffer* tab.

For correct MUC functionality it is necessary to predefine requests first and then initiate MUC by the *Start* button in the *wM-Bus MUC* tab. Then the MUC is switched to th RX mode (MCU sleep, RF IC RX).

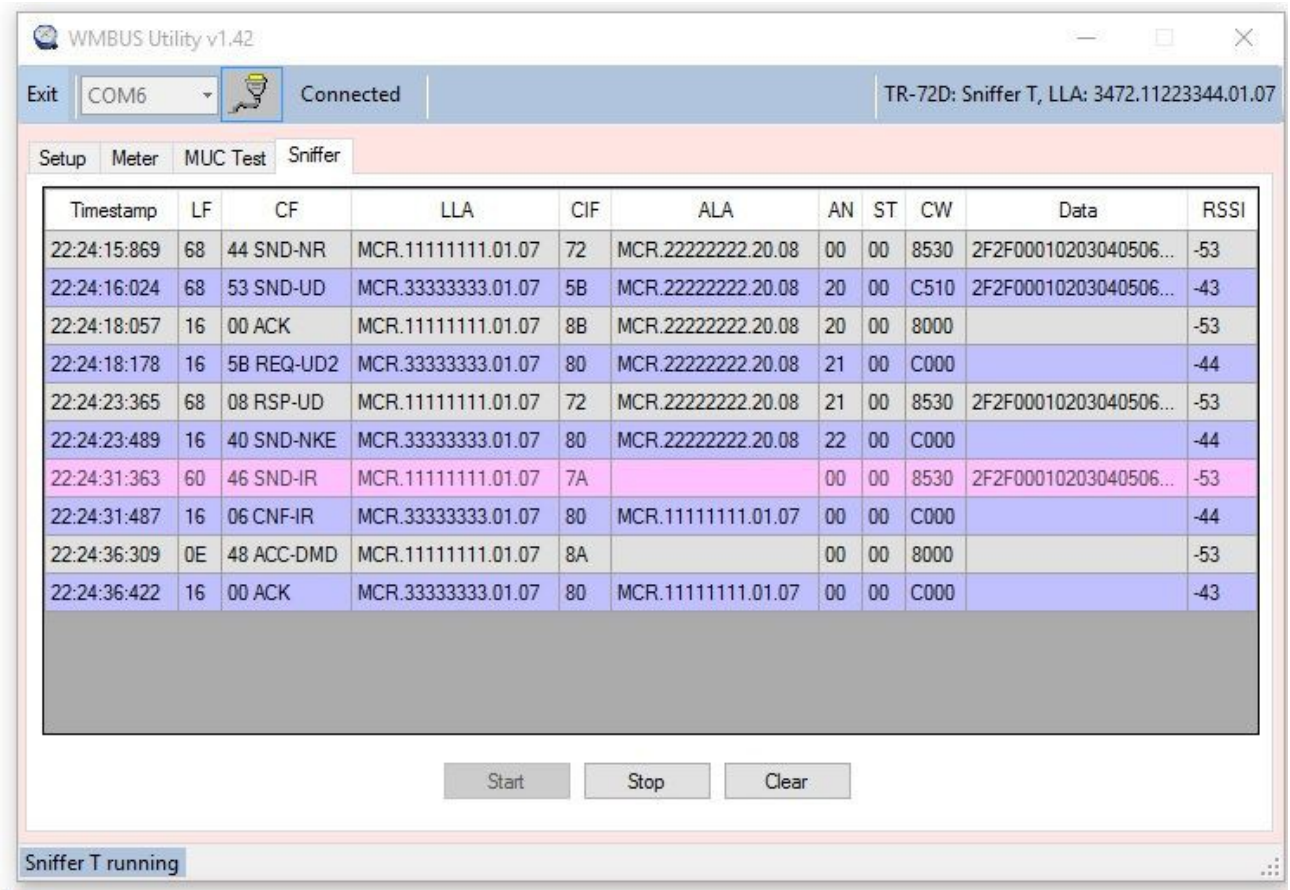
Supported packet types:

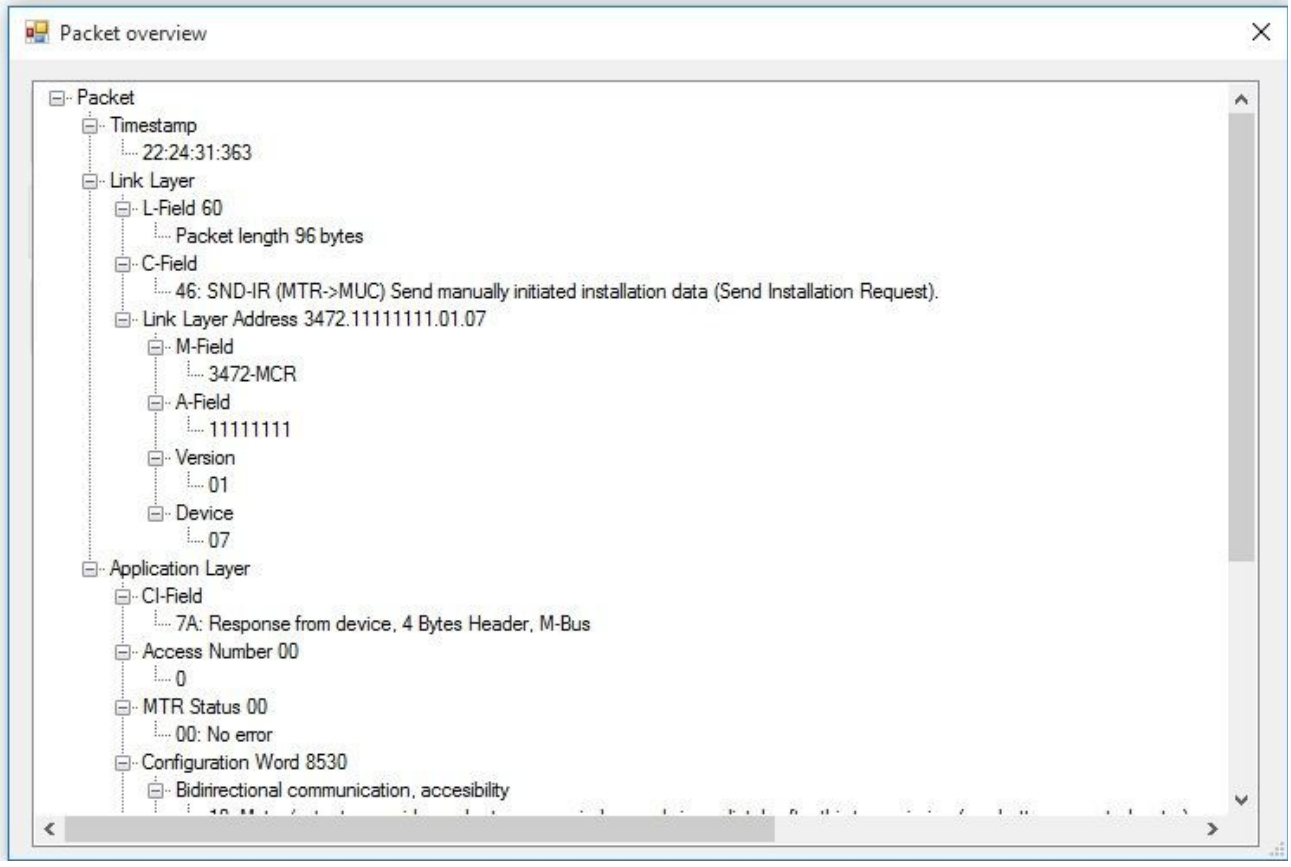
SND-UD (Send command – Send User Data)
CNF-IR (Confirms succesful registration of this meter/actuator into this MUC)
REQ-UD1 (Alarm request – Request User Data Class 1)
REQ-UD2 (Data request – Request User Data Class 2)
ACK (Acknowledge of reception of ACC-DMD)
SND-NKE (Link reset after communication – FAC termination)



wM-Bus Sniffer

By clicking the *Start* button in *wM-Bus Sniffer* tab the TR module is switched to the RX mode (MCU sleep, RF IC RX). Received packets are logged and displayed. Double-click opens a detailed view for given packet.





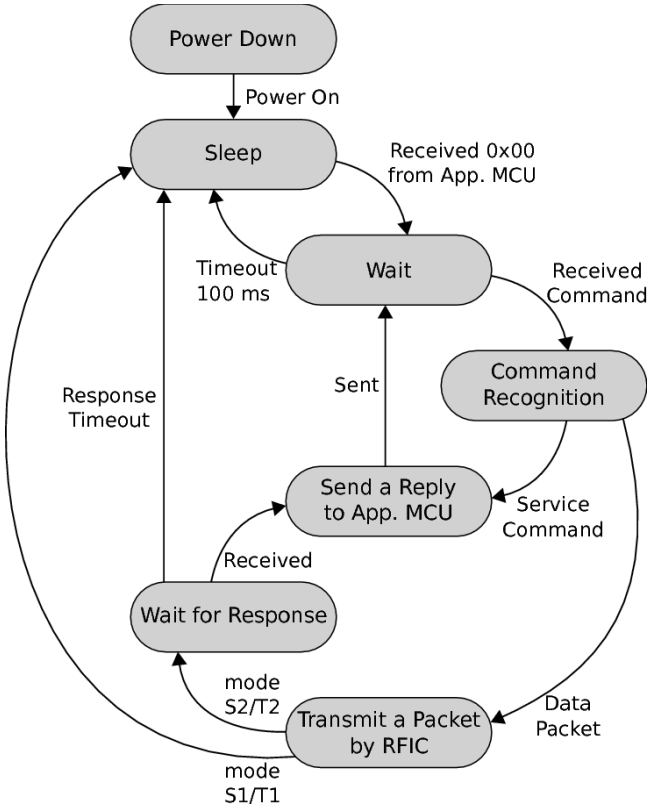
TR module communication states

TR module supports the sleep mode with very low power consumption. The sleep mode is entered automatically after power up.

TR module can be woken up from sleep by sending the `0x00` character on the UART interface. Then, 5 ms after receiving this wake up character, the TR module is ready to receive commands. Any command received during the 100 ms period is processed, the response message is sent back to the UART and TR module is ready for next command. If no command is received during the 100 ms period the TR module is set to sleep again.

If the TR module is in meter communication mode S1 or T1, the TR module will enter the sleep mode immediately once the wM-Bus packet transmission is completed.

If the command from application MCU includes a request for MUC (meter communication mode S2 or T2), the RF circuit will stay in RX mode for a time dedicated to the answer. Throughout the time waiting for an answer, the rest of TR module is in the sleep mode. When the TR module receives an answer from MUC, the TR module will leave the sleep mode, send the answer to the application MCU and wait 100 ms for next command.



UART interface

TR module has a UART interface for serial communication and configuration. The asynchronous UART interface consists of RXD, TXD and GND lines. The settings of UART are as follows:

Baud rate	19200 Bd
Data bits	8
Parity	none
Stop bit	1
Flow control	none

Table 1 - UART settings

Service commands

Table 2 - Supported service commands

Service command format

Service protocol is used to set configuration parameters of the module. Every service command begins with the "`>`" character. Every answer and status response message begins with the "`<`" character. It allows easy orientation in directions if PC terminal is used. Every packet is terminated with the [`CR`] character (`0x0D`). Each packet must be preceded by the wake-up character `0x00` and 2 ms pause.

Input command

`>[cc][rw][data] [CR]`

`[cc]` – Command code

`[rw]` – Read / write symbol

Command	Character	ASCII
Read	?	0x3F
Write	:	0x3A

Table 3 - R/W symbols

`[data]` – Data payload

Response answer

`<[data][CR]`

Response messages

`<OK[CR]`

`OK` – Command received successfully

`<ERR1[CR]`

`ERR1` – Syntax error

`<ERR2[CR]`

`ERR2 – Incorrect input value`

Configuration word

Service commands utilize the Configuration word containing important information regarding wM-Bus device condition. Bits 2 and 3 can be modified by the `06` command, bit 13 by the `07` command, the others are read-only from the user's point of view.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bidirectional communication	Accessibility	Synchronous	Reserved	Mode bit3	Mode bit2	Mode bit1	Mode bit0	Number of encrypted blocks	Number of encrypted blocks	Number of encrypted blocks	Number of encrypted blocks	Content of telegram	Content of telegram	Hop counter	Hop counter

Table 4 - Configuration word

Detailed service command description

00 – wM-Bus Communication mode setup

The TR module can be set to the Meter, Sniffer or MUC mode. Each mode has different communication settings.

Input command

```
>00:[MM][CR]
```

`[MM]` – Selected wireless mode

Input value	Mode
10	Meter - S1
11	Meter - T1
12	Meter - S2
13	Meter - T2
22	MUC - S2 (test)
23	MUC - T2 (test)
40	Sniffer - S
41	Sniffer - T

Table 5 - Available Wireless M-Bus modes

Response message

```
<OK[CR]
```

Example 1

```
>00:10[CR]                    // Switch to Meter S1 mode
<OK[CR]                       // Command confirmed
```

Example 2

```
>00:13[CR]                    // Switch to Meter T2 mode
<OK[CR]                       // Command confirmed
```

Example 3

```
>00:22[CR]                    // Switch to MUC S2 mode
<OK[CR]                       // Command confirmed
```

Example 4

```
>00:41[CR]                    // Switch to Sniffer T mode
<OK[CR]                       // Command confirmed
```

Example 5

```
>00?[CR]                      // Request for current mode
<41[CR]                       // Answer: Sniffer T
```

01 – Link address

The address field of the data link layer always includes the sender address. The link layer protocol supports the unique 8 byte device identification consisting of a 2 byte manufacturer identification, the 8 digit (4 byte) BCD coded identification number, a one byte version and a one byte device type identification. The address is stored in the TR module EEPROM memory and included in every wM-Bus packet.

Input command

```
>01:[MMMM][IIIIIIII][VV][DD][CR]
```

`[MMMM]` – Manufacturer code, 2 bytes in HEX big endian order, input string range: `0000` .. `ffff`

`[IIIIIIII]` – Serial number, 4 bytes in BCD big endian order, input string range: `00000000` .. `99999999`

`[VV]` – Version number, 1 byte in HEX (`00` .. `ff`)

`[DD]` – Device code, 1 byte in HEX (`00` .. `ff`)

Response message

```
<OK[CR]
```

Example 1

```
>01:5336443322111037 [CR]          // 01: Link address requested
                                   // Manufacturer code: 0x3653
                                   // Serial number: 11223344
                                   // Version: 0x10
                                   // Device type: 0x37
<OK[CR]                            // Command confirmed
```

Example 2

```
>01:abcd67452301abff[CR]           // 01: Link address requested
                                   // Manufacturer code: 0xcdab
                                   // Serial number: 01234567
                                   // Version: 0xab
                                   // Device type: 0xff
<OK[CR]                            // Command confirmed
```

Example 3

```
>01?[CR]                           // 01? Link address requested
<abcd67452301b1cf[CR]              // Answer:
                                   //   Manufacturer code: 0xcdab
                                   //   Serial number: 01234567
                                   //   Version: 0xb1
                                   //   Device type: 0xcf
```

02 – Application address

Packets have included either short of long header. See the *OMS Specification* [2], pages 20, 52 and 53. If a packet is provided with long header, the address of application layer (ALA) is included in it. See examples in the *OMS Specification* [2], pages 71, 72, 76 and 77.

The application address can be set by the user manufacturing Meters utilizing TR-72D-WMB modules via servivice command.

Input command

`>02:[MMMM][IIIIIIII][VV][DD][CR]`

`[MMMM]` – Manufacturer code, 2 bytes in HEX big endian order, input string range: `0000` .. `ffff`

`[IIIIIIII]` – Serial number, 4 bytes in BCD big endian order, input string range: `00000000` .. `99999999`

`[VV]` – Version number, 1 byte in HEX (`00` .. `ff`)

`[DD]` – Device code, 1 byte in HEX (`00` .. `ff`)

Response message

`<OK[CR]`

Example 1

```
>02:01cd15948723aa3c[CR]          // Application address
                                  // Manufacturer code: 0xcd01
                                  // Serial number: 23879415
                                  // Version: 0xaa
                                  // Device type: 0x3c
<OK[CR]                           // Command confirmed
```

Example 2

```
>02?[CR]                          // Application address
<03cb15248743ac3d[CR]             // Answer:
                                  //   Manufacturer code: 0xcb03
                                  //   Serial number: 43872415
                                  //   Version: 0xac
                                  //   Device type: 0x3d
```

03 – AES key

The CBC (Cipher Block Chaining) encryption for AES128 uses a 128 bit (16 B) initialization vector to start the encryption of the first block. The AES key is stored in the TR module EEPROM memory.

Input command

```
>03:[KKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK][CR]
```

[K_1..K_{32}] – AES key, 16 bytes in HEX big endian order, input value range: 00..00 to ff..ff

Response message

```
<OK[CR]
```

Example 1

```
>03:0102030405060708090a0b0c0d0e0f[CR]
                              // AES key:[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
<OK[CR]                       // Command confirmed
```

Example 2

```
>03?[CR]                      // Requested actual AES key
<0102030405060708090a0b0c0d0e0f[CR]
                              // Answer: [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
```

04 – wM-Bus Access number

The access number combined with the transmitter address is used to identify a telegram. The meter access number is generated by TR module and can be changed by application MCU. The TR module increments the access number by 1 with every synchronous transmission. Asynchronous transmissions apply the access number of the last synchronous transmission.

Input command

```
>04:[AA][CR]
```

[AA] – Access number (1 byte) in HEX, input value range: 00 .. ff

Response message

```
<OK[CR]
```

Example 1

```
>04:01[CR]                    // Set access number to 0x01
<OK[CR]                       // Command confirmed
```

Example 2

```
>04?[CR]                      // Access number
<01[CR]                       // Answer: 0x01
```

05 – wM-Bus Status byte

Input command

```
>05:[SS][CR]
```

`[SS]` – Status byte, 1 byte in HEX, input string range: `00` .. `ff`

Response message

```
<OK[CR]
```

Example 1

```
>05:0a[CR]                          // Set status byte to 0x0a
<OK[CR]                             // Command confirmed
```

Example 2

```
>05?[CR]                            // Status byte
<0a[CR]                             // Answer: 0x0a
```

06 – wM-Bus Contents of meter telegram

Input command

```
>06:[CC][CR]
```

`[CC]` – Contents of meter telegram (bit 2 and bit 3 of the Configuration Word, see Table 4 - Configuration word)

Input value	Bit 3	Bit 2	Contents of the telegram
00	0	0	Standard data telegram with unsigned variable meter data
04	0	1	Signed data telegram (consists of meter data with a signature approved for billing)
08	1	0	Static telegram (consists of parameter, OBIS definitions and other data points which are not frequently changed)
0c	1	1	Reserved - not implemented

Table 6 - Meter telegram format

Response message

```
<OK[CR]
```

Example 1

```
>06:04[CR]                    // Request for signed data telegram
<OK[CR]                       // Command confirmed
```

Example 2

```
>06?[CR]                      // Request for current telegram contents
<04[CR]                       // Answer: signed data telegram
```

07 – wM-Bus Transmit mode

Input command

```
>07:[MM][CR]
```

`[MM]` - Transmit mode selection byte (bit 13 of the Configuration Word, see Table 4 - Configuration word)

Input value	Bit 13	
00	0	Asynchronous transmission
20	1	Synchronous transmission

Table 7 - Transmit selection byte

Response message

```
<OK[CR]
```

Example 1

```
>07:00[CR]                    // Request for asynchronous transmission
<OK[CR]                       // Command confirmed
```

Example 2

```
>07:20[CR]                    // Request for synchronous transmission
<OK[CR]                       // Command confirmed
```

Example 3

```
>07?[CR]                      // Request for current transmission mode
<20[CR]                       // Answer: synchronous transmission
```

08 – Transmit power

The TR module transmit power setup.

Input command

`>08:[PP][CR]`

`[PP]` – Transmit power

Input string	Transmit power
00	-30 dBm
01	-24 dBm
02	-12 dBm
03	-6 dBm
04	0 dBm
05	+5 dBm
06	+10 dBm
07	+12 dBm

Table 8 - transmit power levels

Response message

`<OK[CR]`

Example 1

```
>08:05[CR]                    // Requested +5 dBm power level
<OK[CR]                       // Command confirmed
```

Example 2

```
>08?[CR]                      // Requested actual power level
<02[CR]                       // Answer: -2 dBm
```

09 – Module information

The TR module version check.

Input command

`>09?[CR]`

Response message

`<[HW][FW][CR]`

`[HW]` - Hardware (PCB) version, output string format: `x.xx`

`[FW]` - Firmware version, output string format: `x.xx`

Example 1

```
>09?[CR]                      // Requested module information
<1.031.26[CR]                       // Answer:
                              //   HW version 1.03
                              //   FW version 1.26
```

0A – Module control

The sleep mode activation / deactivation (for MUC and Sniffer only) and TR-72D-WMB restart.

Input command

```
>0A:[SM][CR]
```

`[SM]` – sleep mode

Input value	Command
00	Go to sleep mode
01	Wake up to RX mode
02	Restart TR-72D-WMB

Table 9 - module control strings

Response message

```
<OK[CR]
```

Example 1

```
>0A:00[CR]                    // Switch to sleep mode
<OK[CR]                       // Command confirmed
```

Example 2

```
>0A:01[CR]                    // Wake-up requested
<OK[CR]                       // Command confirmed
```

Example 3

```
>0A?[CR]                      // TR-72D-WMBB mode requested
<01[CR]                       // Module is in RX mode
```

Example 4

```
>0A:02[CR]                    // TR-72D-WMB restart requested
<OK[CR]                       // Immediate restart follows
```

0B – Set MUC flags

MUC control (for MUC only).

Input command

```
>0B:[MC][CR]
```

`[MC]` – MUC control:

bits 7, 6, 5 and 4: Reserved for future use, must be cleared (0)

bit 3: Transmitting of predefined request `level0` enable. (1 – enabled, 0 – diasbled)

bit 2: Transmitting of predefined request `level1` enable. (1 – enabled, 0 – diasbled)

bit 1: CNF-IR packet transmitting (Meter response to SND-IR) enable (1 – enabled, 0 – diasbled)

bit 0: ACK transmitting (Meter response to ACC-DMD) enable (1 – enabled, 0 – diasbled)

Example 1

```
>0B:0F[CR]                    // Enabled  level0, level1, CNF-IR, ACK
<OK[CR]                       // Command confirmed
```

0C – Battery check

Checking the state of the battery.

Input command

```
>0C?[CR]
```

Response message

```
<[BS][CR]
```

[BS] – Battery status:

0x00: Battery O.K., voltage > 2.95 V

0xff: Battery low, voltage < 2.95 V

Example 1

```
>0C?[CR]                          // Request for battery check
<00[CR]                           // Answer: battery O.K.
```

Example 2

```
>0C?[CR]                          // Request for battery check
<ff[CR]                           // Answer: battery exhausted
```

0D – Read device ID

Reading the factory set serial number identifying the device.

Input command

```
>0D?[CR]
```

Response message

```
<[NNNNNN][CR]
```

[NNNNNN] – Serial number, six digits in hexadecimal

Example

```
>0C?[CR]                          // Request for serial number
<1234AB[CR]                       // Answer: serial number = 1193131
```

Data commands

Code	Command name	Page
80	wM-Bus Message transmission	12
81	wM-Bus Prepare MUC request	13

Table 10 - Supported data commands

Data command format

Data ready to be sent via the wM-Bus is included in a packet, containing the header, packet length, command number and checksum byte. The whole data packet is in **binary** format. Each packet must be preceded by the **wake-up** character `0x00` and 2 ms pause.

Input packet

`[head][LEN][CMD][data][CRC]`

`[head]` – First byte is always value `0xff`

`[LEN]` – Total length of all bytes after the `[LEN]` section including the checksum

`[CMD]` – Command identification byte

`[data]` – M-Bus data payload ready to send

`[CRC]` – The exclusive-or of all bytes after the `[head]` and prior to the checksum

Response packet

`[LEN][result][data][CRC]`

`[LEN]` – Total length of all bytes after the `[LEN]` section including checksum

`[result]` – Result of the transaction

`[data]` – M-Bus data payload received

`[CRC]` – The exclusive-or of all bytes after the `[head]` and prior to the checksum

Detailed data command description

80 - wM-Bus Message transmission

The transmission of the wM-Bus message.
When the TR module receives this command, check the settings, creates and transmits a wM-Bus message. If the settings check fails, the error code is sent to the application MCU and wM-Bus message is not sent.

Input packet

`0xff[LEN]0x80[data={CF,CIF,AES_BC,USER_DATA}][CRC]`

`0xff` – First byte is always value 0xff

`[LEN]` – Total length of all bytes after the `[LEN]` section including checksum

`0x80` – Command identification byte

`[data]` – M-Bus data payload ready to send

`[CF]` – The C-field declares the message type according to *OMS specification* [2]

`[CIF]` – The CI-field indicates the main telegram function and the type of coding

`[AES_BC]` – Number of encrypted blocks, input range: 0 .. 5 (0 = unencrypted)

`[USER_DATA]` – Data payload

`[CRC]` – The exclusive-or of all bytes after the `0xff` and prior to the checksum

Response message

`[LEN][result][data][CRC]`

`[LEN]` – Total length of all bytes after the `[LEN]` section including checksum

`[result]` – Result of the transaction

Value	Result of the transaction
`0xff`	RFIC error - internal communication error
`0xfe`	MUC timeout (for S2 and T2 mode only)
`0xfd`	Manchester codec error in received message (for S2 and T2 mode only)
`0xfc`	CRC error in received message (for S2 and T2 mode only)
`0xfb`	Unsupported CF
`0xfa`	Unsupported CIF
`0xf9`	Too many encryption blocks (`AES_BC > USER_DATA/16 + 1`)
`0xf6`	The received message is too long (only for S2 and T2 mode)
`0xf5`	The received message encryption is not supported (only for S2 and T2 mode). The TR module supports the encryption mode 5 only.
`0xf3`	UART CRC mismatch
`0x00`	The message was sent successfully, the answer was successfully received

Table 11 - Message transmission result codes

`[data]` – Received M-Bus data payload

`[CRC]` – The exclusive-or of all bytes after the `0xff` and prior to the checksum

Example 1 – Unidirectional meter

Application MCU sent:

```
0x1580447A00000102030405060708090a0b0c0d0e0f23
```

`0xff` = Header byte

`[LEN]` = `0x15` (`[CMD]` + `[data]` + `[CRC]` = 21 bytes)

`[CMD]` = `0x80`

`[CF]` = `0x44` (SND-NR)

`[CIF]` = `0x7A` (Response from device)

`[AES_BC]` = `0x00` (No encryption)

`[USER_DATA]` = 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f

`[CRC]` = `0x23` (`[LEN]` xor `[CMD]` xor $[data_1]$ xor ... xor $[data_n]$)

The TR module checks CRC, CF, CIF, AES blocks, creates a wM-Bus message and transmit it. The TR module is set to unidirectional meter mode S1 or T1 and does not wait for an answer from MUC.

Application MCU receives:

```
0x020002
```

`[LEN]` = `0x02` (`[result]` + `[DATA]` + `[CRC]` = 2 bytes)

`[result]` = `0x00` (The message was sent successfully)

`[data]` = `null` (No data was received)

`[CRC]` = `0x02` (`[LEN]` xor `[result]`)

Example 2 – Sniffer

Application MCU receives:

```
0x2400204472341111111101077A000010852F2F000102030405060708090A0B0C0D0E0F9976
```

The format is the same as for response message.

`[LEN]` = `0x24` (`[result]` + `[data]` + `[CRC]` = 36 bytes)

`[result] = 00` Packet O.K. (Other values – errors, see Table 11)

```
[DATA] = 20 44 72 34 11 11 11 11 01 07 7A 00 00 10 85 2F 2F
         F0 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 99
```

Received wM-Bus data, without CRC of individual blocks, decrypted
The last data byte is RSSI – the value from the RF IC: `[RSSI] = 0x99`
The utility converts this to dBm. Linear conversion is used:

`RSSI [dBm] = RSSI_LEVEL/2 - 130`

Thus, `RSSI [dBm] = 153/2 – 130 dBm = -53.5 dBm`

`[CRC] = 0x76 ([LEN] xor [result] xor` $[data_1]$ `xor ... xor` $[data_n]$`)`

81 - wM-Bus Prepare MUC request

Applies to MUC only. This command prepares MUC request for a Meter. The first part of the command (level 0) sets the initial message to enter the FAC. The second part of the command (level 1) sets the request for the Meter. The FAC is automatically exited by sending the SND-NKE message. Each packet must be preceded by the **wake-up** character `0x00` and 2 ms pause.

Input packet

`0xff[LEN]0x81[Level][data={CF,CIF,AES_BC,USER_DATA}][CRC]`

`0xff` – First byte is always the `0xff` value

`[LEN]` – Total length of all bytes after the `[LEN]` section including the checksum

`0x81` – Command identification byte

`[Level]` – Two level request:

Value	Message type
0x00	Reply to SND-NR to initiate FAC
0x01	Request message for a Meter

Table 12 - two level reply messages from MUC

`[data]` – M-Bus data payload ready to send

`[CF]` – The C-field declares the message type according to the *OMS specification* [2]

`[CIF]` – The CI-field indicates the main telegram function and the type of coding

`[AES_BC]` – Number of encrypted blocks, input range: 0 .. 5 (0 = unencrypted)

`[USER_DATA]` – Data payload

`[CRC]` – The exclusive-or of all bytes after the `[head]` and prior to the checksum

Response message

`[LEN][result][CRC]`

`[LEN]` – Total length of all bytes after the `[LEN]` section including the checksum

`[result]` – Result of the transaction:

Value	**Result of the transaction** (return code)
`0x00`	O.K.
`0xf6`	Number of user data. USER_DATA > 5 blocks (80 B)
`0xf9`	Number of encrypted blocks. > (USER_DATA_LEN + 0x01), see the 0x80 command description
`0xfb`	Illegal C-Field
`0xfa`	Illegal CI-Field
`0xf3`	UART CRC mismatch
`0xf8`	Illegal LEVEL (legal values are <0;1>)

Table 13 - message transmission result codes

`[data]` – Received M-BUS data payload

`[CRC]` – The exclusive-or of all bytes after the `[head]` and prior to the checksum

Example 2 (bidirectional meter)

Application MCU sent:
`0xff1580447A000001020304050607080 90a0b0c0d0e0f23`

`[head]` = `0xff` header byte

`[LEN]` = `0x15` (`[CMD]` + `[data]` + `[CRC]` = 21 bytes)

`[CMD]` = `0x80`

`[CF]` = `0x44` (SND-NR)

`[CIF]` = `0x7A` (Response from device 4 bytes header)

`[AES_BC]` = `0x00` (No encryption)

`[USER_DATA]` = `00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f`

`[CRC]` = `0x23` (`[LEN]` xor `[CMD]` xor $[data_1]$ xor ... xor $[data_n]$)

The TR module checks CRC, CF, CIF, AES blocks, creates a wM-Bus message and transmit it. The TR module is set to bidirectional meter mode S2 or T2 and waits for an answer from MUC. The answer is sent to application MCU.

Application MCU receives from MUC:
`0x1900167A2448462705770107800100000024480107231A00C01E`

`[LEN]` = `0x19` (`[result]` + `[DATA]` + `[CRC]` = 25 bytes)

`[result]` = `0x00` (The message was successfully sent, the answer was successfully received)

`[data]` = "`167A2448462705770107800100000024480107231A00C0`" consists of:

L-field = `0x16`

C-field = `0x76` (REQ-UD1)

MUC address = `2448462705770107`

CI-field = `0x80` (Transport layer to device)

Meter address = `0100000024480107`

MUC access number = `0x23`

MUC status (RSSI) = `0x1a`

MUC configuration word = `0xc000` (Unlimited access)

`[CRC]` = `0x1E` (`[LEN]` xor `[result]` xor $[data_1]$ xor ... xor $[data_n]$)

The next Frequent Access Cycle is managed by application processor.

References

[1] *TR-72D-WMB datasheet*
www.iqrf.org/downloads

[2] *Open Metering System Specification, Volume 2, Primary Communication*
www.oms-group.org/download/OMS-Spec_Vol2_Primary_v301.pdf

[3] *Glossary of Terms related to OMS (Open Metering System) Specification*
www.oms-group.org/download/OMS-Spec_Glossary_v101.pdf

[4] GW-USB-06-*WMB User's guide*
www.iqrf.org/downloads

Supported devices

- TR72-D(A/C/T/AT/CT)-WMB
- TR76-D(A)-WMB
- GW-USB-06-WMB

Document history

- 151209 First release.

Sales and Service

Corporate office

MICRORISC s.r.o., Prumyslova 1275, 506 01 Jicin, Czech Republic, EU
Tel: +420 493 538 125, Fax: +420 493 538 126, www.microrisc.com

Partners and distribution

Please visit www.iqrf.org/partners

Quality management

ISO 9001 : 2009 certified

CE

Trademarks

Legal

On-line support: support@iqrf.org

Smarter wireless. Simply.