Wireless M-BUS

Wireless M-BUS (wMbus) is a radio technology for meters and sensors that operates in 433 MHz and 868 MHz with short range of 30 to 150m (depending of environment).

Table of Contents

Radio Parameters

Range

Reception range highly depends on the Environment.

Frequency	Environment	Range	Placement recommendation
868 MHz	Indoor	10-50m	• Vertically: every 2nd to 3rd floor • Horizontally: every 1st to 2nd entrance (2-3 apartments in between) • Additional gateway in basement if needed
868 MHz	Outdoor	150-200m	High position, when possible in Line of sight

RSSI

RSSI values are typically in the range von -60 dBm (good) to -107 dBm (bad)

RSSI Value	
Good Reception (close to gateway)	-60 dBm
Bad Reception (far from gateway)	-107 dBm

Support of Lobaro Devices

Lobaro offers multiple solutions to collect wMbus telegrams and forward them via NB-IoT, LoRaWAN, LAN, etc. called Wireless M-Bus Gateway or Bridge.

Utility meter compatibility

The Lobaro wMBUS Gateways are working with **every meter** using standard 868 MHz wMbus:

- wireless MBUS S1, C1 or T1 mode (unidirectional 868 MHz modes following DIN EN 13757-4)
- Open metering specification (OMS) v3 & v4
- Sensus RF Bubble UP - Manufacturer specific radio protocol

433 MHz is not supported by any Lobaro Gateway yet

We created a detailed **list of over 400 meters** that were received by our gateway in supported-devices.csv

The list includes the following devices:

wMBUS meter	Type	Manufacturer	More information
Q caloric 5.5	Heat cost	Qundis	External Link
Sontex 868	Heat cost	Sontex	External Link
SHARKY 775	Heat	Diehl Metering	External Link
MULTICAL® 603	Heat	Kamstrup	External Link
QALCOSONIC E1	Heat & Cooling	Axioma Metering	External Link
Sonometer 30	Heat & Cooling	Danfoss	External Link
INVONIC H	Heat & Cooling	Apator	External Link
iPERL®	Water	Sensus	External Link
Hydrus	Water	Diehl Metering	External Link
MULTICAL® 21 / flowIQ® 210x	Water	Kamstrup	External Link
Q water 5.5	Water	Qundis	External Link
Modularis yFlow	Water addon	Hermann Pipersberg	External Link
Munia	Temp. & Humidity	Weptech	External Link
Qalcosonic W1 (Honeywell Q400)	Water	Axioma Metering	External Link
EquaScan eHCA	Heat cost	iTron	External Link
EquaScan hMIU	Heat	iTron	
EquaScan pMIO	Pulse Meter	iTron	
EquaScan wMIU	Water meter	iTron	
EquaScan iSD	Smoke Detector	iTron	
ULTRIMIS W	Water	Apator	External Link
V200 / V210 HYBRID	Water	Elster	External Link

Parsing

For parsing wMbus telegrams you can use our wMbus Parsing API

M-Bus Telegram Structure

Storage number

The storage number does group value by points in time.

Storage Number	When?
0	At time of reading
1 .. n	Historic values, e.g. Monthly or Due Date Values

There can be multiple value types inside a single storage number, typical values are:

Value type	Examples	Meaning
Date	31.12.2022	All values of the storage number relate to this time.
Time & Date	01.02.2023 23:55:00	

Meter value	Volume / Energy / Temperature / H.C.A. / etc. e.g. 20.4 °C	The value measured by the meter
Error Value	[0 1 0 0 0 0 0 0]	Ofen manufacturer specific error flags. Mostly used in storage number 0, but can also appear in others.
Any other		Some meters have other value types

Encryption

Wireless M-Bus Telegrams are usually encrypted and must be decrypted using a 16-Bit key.

When received by the Lobaro Plattform, you can configure decryption keys that are applied up on reception (see: wMbus Keys).

Telegram Coding

Telegram type A

- Always for wMBUS S1 mode
- Always for wMBUS T1 mode
- Possible for wMBUS C1 mode
    - The detection is done via the syncword on phy level by the receiver

L-Field

The first byte of the first block is the length field. The field specifies the number of subsequent user bytes, including control and address bytes, but not CRC bytes. If ((L-9) MOD 16) is not zero, then the last block must contain ((L-9) MOD 16) data bytes +2 CRC bytes. All other blocks, except the first block, must always contain 16 data bytes + 2 CRC bytes.

CRCs

- First crc after 10 bytes
- Then every 16 bytes one CRC of length 2 (= 18)
- Plus one crc at the end (unless ending on 16 byte block)

Telegram type B

- Possible for wMBUS C1 mode
    - The detection is done via the syncword on phy level by the receiver

L-Field

The first byte of the first block is the length field. The field sets the number of all subsequent bytes, including all CRC bytes.

CRCs

The link layer check of telegram format B is performed on a maximum of 128 bytes, including the CRC field. Telegrams with a length of up to 128 bytes, including CRC and L field, contain a single CRC field covering both the first and the second block. Telegrams with a length between 131 bytes and 256 bytes (maximum length) contain two CRC fields, with the second CRC field covering the optional block.

JS wMbus Parser

Compatible Parser for the Lobaro Platform, TTN / TTI and Chirpstack.

wMbus Header Parser

```
//////////////////////
// Helper
//////////////////////

// Convert a hex string to a byte array
function hexToBytes(hex) {
    for (var bytes = [], c = 0; c < hex.length; c += 2)
        bytes.push(parseInt(hex.substr(c, 2), 16));
    return bytes;
}

// Convert a byte array to a hex string
function bytesToHex(bytes) {
    for (var hex = [], i = 0; i < bytes.length; i++) {
        var current = bytes[i] < 0 ? bytes[i] + 256 : bytes[i];
        hex.push((current >>> 4).toString(16));
        hex.push((current & 0xF).toString(16));
    }
    return hex.join("");
}

// MFieldToString is Calculated from MField (15 bit) and represented as 3 ASCII chars
// First bit is ignored and used to define the uniqueness of the Meter Id
// AAA = 0x0421 = 0 00001 00001 00001
// ZZZ = 0x6b5a = 0 11010 11010 11010
function MFieldToString(m) {
    var letters = ""
    var char3 = (m&0x1F)+64
    var char2 = ((m&0x3E0) >> 5)+64
    var char1 = ((m&0x7C00) >> 10)+64
    letters += String.fromCharCode(char1)
    letters += String.fromCharCode(char2)
    letters += String.fromCharCode(char3)
    return letters
}


// ParseWmbus returns a struct with the wMbus telegram header information plus the raw telegram
// telegram is the whole telegram as byte slice (each element of the slice contains one byte value as number)
//
// Data Link Layer:
// ----------------
//   0  |   1  |  2   |   3  |  4   |   5  |   6  |   7  |   8  |   9    |  10  |  11  | -> Application Layer
// 0x2E | 0x44 | 0xB0 | 0x5C | 0x12 | 0x13 | 0x00 | 0x00 | 0x02 | 0x1B   | 0x8A |  ... |
// Len  |  C   |  M   |  M   |   ID |  ID  |  ID  |  ID  | Ver  | Device |  CRC |  CRC |
function ParseWmbus(telegram) {
    var id = bytesToHex(telegram.slice(4, 8).reverse());
    var mField = MFieldToString((telegram[2] << 0) + (telegram[3] << 8));
    var version = telegram[8];
    var deviceType = telegram[9];

    return {
        "Id": id,
        "MField": mField,
        "Version": version,
        "Device": deviceType,
        "Telegram": bytesToHex(telegram)
    }
}
```