Protocol of data exchange with modem via USB interface

Version 2016.03.10

- Modem connects to USB-host as USB device of CDC class (virtual COM port in Windows, ttyUSB or ttyACM in Linux)
- Because real RS-232 is not used in this interface, parameters of serial port, opening on the host (baudrate, number of bits, parity, etc) may be anyone
- Data is in binary format
- «Network address» of modem is **0xff**
- Multibyte numbers are transmitting starting from low byte (little endian format)

1. Reading last coordinates pack.

Format of request frame (from host to modem)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of modem	0xff
1	1	uint8_t	Type of packet	0x03
2	2	uint16_t	Code of data in packet	0x4100
4	2	uint16_t	Access mode	0x0000
6	2	uint16_t	CRC-16 (see appendix)	

Format of answer frame (from modem to host)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of modem	0xff
1	1	uint8_t	Type of packet	0x03
2	1	uint8_t	Number of bytes of data transmitting	0x28
3	40 (0x28)	40 bytes	Data structure (see lower)	
43	2	uint16_t	CRC-16 (see appendix)	

Format of data field (40 bytes)

Offset	Size (bytes)	Description
0	36 (6*6)	Six last coordinates structures received by modem (see lower)
36	4	Reserved

Format of coordinates structure (6 bytes)

Offset	Size (bytes)	Description
0	1	Address of device
1	2	Coordinate X, mm (int16_t)
3	2	Coordinate Y, mm (int16_t)
5	1	Byte of flags: Bit 0: 1 – no relevant coordinates (red mode in dashboard) Bit 1: 1 – temporary mobile beacon on frozen map (blue mode) Bit 2: 1 – beacon is used for hedgehog positioning

2. Reading device height

Format of request frame (from host to modem)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of device	0x01…0x63
1	1	uint8_t	Type of packet	0x03
2	2	uint16_t	Code of data in packet	0x5002
4	2	uint16_t	Access mode	0x0002
6	2	uint16_t	CRC-16 (see appendix)	

Format of answer frame (from modem to host)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of device	0x01…0x63
1	1	uint8_t	Type of packet	0x03
2	1	uint8_t	Number of bytes of data transmitting	0x1c
3	28 (0x1c)	28 bytes	Data structure (see lower)	
43	2	uint16_t	CRC-16 (see appendix)	

Format of data field (28 bytes)

Offset	Size (bytes)	Description
0	2	Coordinate Z, mm (int16_t)
2	26	Reserved

3. Reading/writing modem configuration.

3.1 Reading modem configuration.

Format of request frame (from host to modem)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of modem	0xff
1	1	uint8_t	Type of packet	0x03
2	2	uint16_t	Code of data in packet	0x5000
4	2	uint16_t	Access mode	0x0000
6	2	uint16_t	CRC-16 (see appendix)	

Format of answer frame (from modem to host)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of modem	0xff
1	1	uint8_t	Type of packet	0x03
2	1	uint8_t	Number of bytes of data transmitting	0x20 (FW 4.27-) 0x30 (FW 4.28+)
3	0x20 (FW 4.27-) 0x30 (FW 4.28+)	structure	Data structure (see section 3.3)	
0x23 or 0x33	2	uint16_t	CRC-16 (see appendix)	

3.2 Writing modem configuration.

Warning! To write modem configuration you must read configuration, setup the data fields described in following section, and then write it. Do not change any other bytes in structure, this may degrade the work of modem

Format of request frame (from host to modem)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of modem	0xff
1	1	uint8_t	Type of packet	0x10
2	2	uint16_t	Code of data in packet	0x5000
4	2	uint16_t	Access mode	0x0000
6	1	uint8_t	Number of bytes of data transmitting	0x20 (FW 4.27-) 0x30 (FW 4.28+)
7	0x20 (FW 4.27-) 0x30 (FW 4.28+)	structure	Data structure (see section 3.3)	
0x27 or 0x37	2	uint16_t	CRC-16 (see appendix)	

Format of answer frame (from modem to host)

Offset	Size (bytes)	Type	Description	Value
0	1	uint8_t	Address of device	0xff
1	1	uint8_t	Type of packet	0x10
2	2	uint16_t	Code of data	0x5000
4	2	uint16_t	reserved	
6	2	uint16_t	CRC-16 (see appendix)	

3.3 Structure of modem configuration data.

Most fields of data structure are not explained. Do not change these fields! They are useful for adjustment system from dashboard program, unauthorized changing may degrade the work of modem

Offset	Size (bytes)	Type	Description
0	1	uint8_t	Address of starting beacon for building map. Modem begins building map from this beacon if it exist.
1	27	27 bytes	Not explained
28	1	uint8_t	Control flags: Bit 0: 1 = map frozen (freeze map), 0 = map unfrozen (unfreeze map) Bit 1…4: not explained Bit 5: 1 = mirror map (change works only when the map is not frozen) Bit 6: 1= power save mode (power save works only when the map is frozen, set this flag in the same command when freezing map) Bit 7: not explained
29	2	2 bytes	Not explained
31	1	uint8_t	N, determines maximum frequency of retrieving hedgehog coordinates F= 2^(N-1) Hz, N= 0…5
32	0x00 (FW 4.27-) 0x10 (FW 4.28+)	16 bytes or 0 bytes	Not explained

Appendix 1. Calculating CRC-16.

For checksum the CRC-16 is used. Last two bytes of N-bytes frame are filled with CRC-16, applied to first (N-2) bytes of frame. To check data you can apply CRC-16 to all frame of N bytes, the result value should be zero.

Below is the implementation of the algorithm in the 'C'.

```
typedef ushort ModbusCrc;//  ushort – two bytes

typedef union {
        ushort w;
        struct{
                uchar lo;
                uchar hi;
        } b;
        uchar bs[2];
} Bytes;

static ModbusCrc modbusCalcCrc(const void *buf, ushort length)

{
        uchar *arr = (uchar *)buf;
        Bytes crc;
        crc.w = 0xffff;
        while(length--){
                char i;
                bool odd;
                crc.b.lo ^= *arr++;
                for(i = 0; i < 8; i++){
                        odd = crc.w & 0x01;
                        crc.w >>= 1;
                        if(odd)
                                crc.w ^= 0xa001;
                }
        }

        return (ModbusCrc)crc.w;
}
```