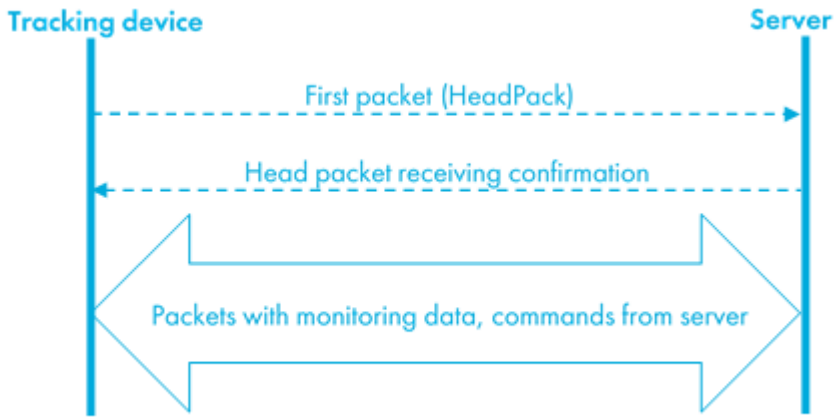
Galileosky protocol supports bi-directional data exchange between the tracking device and the server. The data are transmitted via GPRS channel with the use of TCP/IP protocol. The server must have static address and port for connecting tracking devices as clients.

Data transmission from the tracking device to the server:



After establishing tracking device-server connection the device sends head pack and then main packs with the data. Each pack needs confirmation from the server; if confirmation is not received, the tracking device sends the pack once again.

Packet structure of receiving confirmation

Byte №	Length, byte	Value	Description
1	1	0x02	Header
2 3	2		Checksum of received packet

Note that TCP/IP is a stream protocol, i.e. there are no packets of TCP/IP level for the application server software. Reading from the TPC/IP-socket is reading of the bytes stream but not reading of the packets. The Galileosky protocol packets are not ones of the application level, and for their correct parsing server software has to select a buffer and capture the packet. Do not expect that one reading operation from the socket returns the whole Galileosky protocol packet. The whole Galileosky protocol packet can be received after executing some sequential reading operations, there can be time intervals between them caused by specifics of TCP/IP protocol operation.

First packet

Head packet structure

Byte №	Bit №	Length, byte	Value	Description
1		1	0x01	Header
2	8 7	2	L	Packet length

	6			
	5			
	4			
	3			
	2			
	1			
3	8*			
	7			
	6			
	5			
	4			
	3			
	2			
	1			
4		1		Tag 1
5				Tag 1 data
		1		Tag N
				Tag N data
		1	0xFE	Tag, showing the presence of extended tags
				Extended tags data
L+1		2		Checksum
L+2				

* Indicator of unsent data to the archive: 0 – no; 1 – yes.

Extended tags data structure

Byte №	Bit №	Length, byte	Value	Description
1		2		Length of extended tags data
2				
3		2		Extended tag 1
4				
5				Extended tag data 1
		2		Extended tag N
				Extended tag data N

A high-order bit is an indicator of not transferred data in the archive, 15 low-order bits are the number of bytes in the packet. Maximum packet length is 1000 bytes.

Packet length is calculated from the head tag to checksum beginning. Tags are in ascending order. The data and the checksum are transferred in little-endian format (lower bytes are the first). The checksum is calculated for the whole packet including the header, length field and indicator of unsent data. The checksum is calculated by CRC-16 Modbus algorithm, you can find an example of its realization at this link.

Example of the head packet in hexadecimal form in receiving order. Tags are in accordance with default settings.

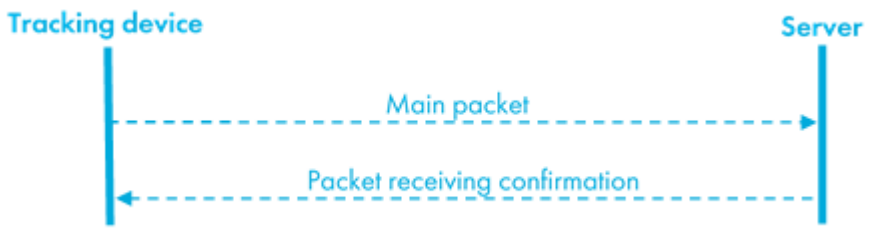
01 20 00 01 9A 02 18 03 38 36 31 32 33 30 30 34 33 39 30 37 36 32 36 04 32 00 FE 06 00 01 00 00 00 00 00 8F 29

Decipherment:

- **01** – header
- **20 00** – length, high-order bit – there are unsent data, in case of masking it, 32 bytes length is received
- **01** – tag 01 – device type

- **9A** – tag value 01 = 154 – Galileosky 7x Plus ext
- **02** – tag 02 – firmware version
- **18** – value of 02 tag = 24
- **03** – tag 03 – IMEI
- **38 36 31 32 33 30 30 34 33 39 30 37 36 32 36** – value of 03 tag – «861230043907626»
- **04** – tag 04 – device number, can be set in settings
- **32 00** – value of 04 tag = 50
- **FE** – sign of extended tags presence (missing if no extended tags are selected)
- **06 00** – length of extended tags – length 6 bytes
- **01 00** – extended tag number
- **00 00 00 00** – extended tag data
- **8F 29** – checksum

Main packet



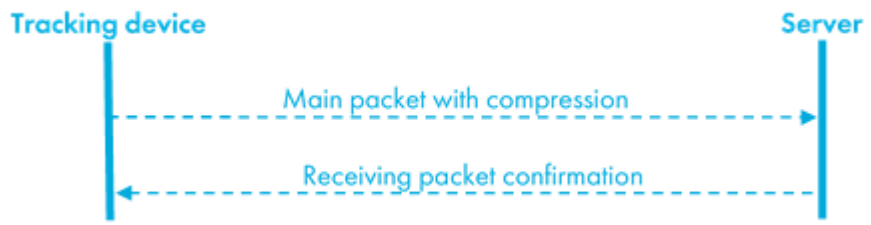
The main pack structure is the same as the structure of the head pack. Main pack may transmit several records from the archive. First record tags go first, then the second record tag and etc.

The data may be coded; XTEA3 algorithm is used for coding with block length 128 bit, key length 256 bit and 32 rounds.

In this case, the header, length and the unsent data indicator stay unchanged, while archives records with the tags are coded. If the data length is not multiple to code block length, missing place is filled with zeros and then coded. The checksum is calculated for coded data packet.

Packet will be transmitted again if its checksum does not correspond to the checksum in the confirmation packet.

Main packet with compression



Depending on settings, the tracking device can transmit data in the main packet with compression. Several records from the archive can be transmitted, structure of the first record differs from the next ones. The first record may contain **minimal data set** (structure of 10 bytes), tags list and tags data. If the first record contains **minimal data set**, every other record also contains it. If the first record has tags list, each other record has tags data in accordance with this list. At the same time, there is the tags list only in the first record. If there are less than 32 tags in the list, tags numbers are transmitted, or bit mask, where each position conforms to a tag number.

Structure of the main packet with compression

Byte №	Length, byte	Value	Description
1	1	0x08	Header
2	2	L	Packet length
3			
4	10		Minimal data set 1
…			
13			
14	2-33		Tags list 1
…			
			Tags data 1
	10		Minimal data set 2
			Tags data 2
			…
	10		Minimal data set N
			Tags data N
L+1	2		Checksum
L+2			

Structure of Minimal Data Set

Byte №	Bit №	Length, bit	Value	Description
1	8	1	0	
	7	25		Date and time

	…			
	1			
2				
3				
	8			
	7			
	6	1		Coordinates validity: 0 – valid, 1 – non-valid
4	5	22		Longitude
	4			
	3			
	2			
	1			
5				
6				
7	8			
	7			
	…			
	1			
8		21		Latitude
9	8			
	…			
	3			

	2	1		Alarm: 0 – no, 1 – yes
	1	9		User tag 0 data
10				

Date and time in **minimal data set** are transmitted in seconds, starting from 00:00:00 of the 1st of January. Year is not transmitted, as it is set in accordance with the current year of the server.

Longitude is transmitted as a whole number without a sign. Value in degrees is calculated by the following formula, where L is a transmitted value in the packet:

$$\text{Longitude} = \frac{360 \times \text{L}}{4194304} - 180$$

Received negative longitude values correspond to Western Hemisphere, positive values-to Eastern one.

Latitude is transmitted as a whole number without a sign. Value in degrees is calculated by the following formula, where L is a transmitted value in the packet:

$$\text{Latitude} = \frac{180 \times \text{L}}{2097152} - 90$$

Received negative longitude values correspond to Southern Hemisphere, positive values-to Northern one.

One bit of transmitted coordinates is approximately equal to 0,00008583 degrees.

Structure of tags list, if they are less than 32

Byte №	Bit №	Length, bit	Value	Description
1	8		1	
	7			
	6			
	5		N	Tags number
	4			
	3			

	2			
	1			
2		1		Tag 1
…				
1+N		1		Tag N

Structure of tags list, if they are more than 31

Byte №	Length, byte	Value	Description
1	1	0xFF	Header
2	32		Bit tags mask
…			
33			

When 0x5C tag is being transmitted (tire-pressure management system PressurePro), there could be recorded either 68 bytes according to the description or 2 bytes – when 0x00FF data are not present – in the tag data.

The data may be coded; XTEA3 algorithm is used for coding with block length 128 bit, key length 256 bit and 32 rounds.

In this case, the header, length and the unsent data indicator stay unchanged, and archives records with the tags are coded. If the data length does not multiple to code block length, missing place is filled with zeros and then coded. The checksum is calculated for coded data packet.

Packet will be transmitted again if its checksum does not correspond to the checksum in the confirmation packet.

Main packet with compression and extended tags

If there is a tag in the tag list or in the bitmask that is responsible for the presence of extended tags, then the list of extended tags is followed by the tag data, then the extended tag data.

Extended tags packet structure in the protocol with compression

Byte №	Length, bytes	Value	Description
1	1	0x08	Header
2	2	L	Packet length

3			
4	10		Minimum data set 1
…			
13			
14	2-33		Tag list 1
…			
	4-8192		Extended tag list 1
			Tag data 1
			Extended tags data 1
	10		Minimum data set 2
			Tag data 2
			Extended tags data 2
			…
			Minimum data set N
			Tag data N
			Extended tags data N
L+1	2		Checksum
L+2			

An extended tag list can be an enumeration of tags or a bitmask. The list representation that is used in the package is determined by the Length parameter in the The structure of extended tag list length table.

Extended tag list structure while using the tag list

Byte №	Length, bytes	Value	Description
1	2	N	List length (tags amount)

2			
3	2		Extended tag 1
4			
5	2		Extended tag 2
6			
7			…
	2		Extended tag N

Extended tag list structure while using a bitmask

Byte №	Length, bytes	Value	Description
1	2	N | 0x8000	List length (bitmask length, bytes)
2			
3	N		Extended tag bitmask
…			
N+2			

The structure of extended tag list length

Byte №	Bit №	Length, bytes	Value	Description
1		2		Tags amount or the bitmask length
2	8		0 – tag list is used 1 – bitmask is used	
	7			
	6			

	5			
	4			
	3			
	2			
	1			

Extended tags data structure in the protocol with compression

Byte №	Bit №	Length, bytes	Value	Description
1		2		Extended tag data length
2				
				Extended tag 1 data
				Extended tag 2 data
				Extended tag N data

Example of an extended tag packet with compression and with tags listed:

08 15 00 82 04 FE 02 00 01 00 FA 00 32 00 08 00 00 00 00 00 00 00 00 00 59 93

08 - header

15 00 – packet length 0x0015 = 21 bytes

82 – masked tags amount 0x80 = 2 tags

04 – 04 tag – device ID

FE – FE tag – extended tags are present

20 00 – extended tags amount 0x0002 = 2 extended tags

01 00 – extended tag 0001

FA 00 - extended tag 00FA

32 00 – tag data 04 0x0032 = 50

08 00 – extended tag data length 0x0008 = 8 bytes

00 00 00 00 – extended tag 0001 value

00 00 00 00 – extended tag 00FA value

59 93 – checksum

Example of an extended tag packet with compression and with bitmask used:

08 12 00 82 04 FE 01 80 06 32 00 08 00 00 00 00 00 00 00 00 00 52 78

08 - header

12 00 – packet length 0x0012 = 18 bytes

82 – masked tags amount 0x80 = 2 tags

04 –04 tag – device ID

FE – FE tag – extended tags are present

01 80 – bitmask length of extended masked tags 0x8000 = 1 byte

06 – extended tags bitmask 00000110 = 0001 and 0002 tags

32 00 – 04 tag data 0x0032 = 50

08 00 – extended tags data length 0x0008 = 8 байт

00 00 00 00 – 0001 extended tag value

00 00 00 00 – 0002 extended tag value

52 78 – checksum

Packet with commands to the tracking device

Server can send a command to device. After receiving and running it, the tracking device sends a packet with reply text.

Structure of packet with command

Byte	Value	Length, byte	Description
1	0x01	1	
2 3	L	2	Packet length
4	0x03	1	Tag
5 … 19		15	IMEI

20	0x04	1	Tag
21 22		2	Tracking device number
23	0xE0	1	Tag
24 … 27		4	Command number
28	0xE1	1	Tag
29	N	1	Command length
30 … 30+N		N	Command text (CP1251)
L+1 L+2		2	Checksum

Checksum is calculated for the whole packet, starting with the header. Command number is a random number set by the server.

Example of command in hexadecimal form in the order of receiving:

01 20 00 03 38 36 38 32 30 34 30 30 35 36 34 37 38 33 38 04 00 00 E0 00 00 00 00 E1 06 73 74 61

74 75 73 50 22

Decipherment:

- **01** – header
- **20 00** – length of 32 bytes
- **03** – tag 03 – IMEI
- **38 36 38 32 30 34 30 30 35 36 34 37 38 33 38** – value of 03 tag – «868204005647838»
- **04** – tag 04 – device number, set in settings
- **00 00** – value of 04 tag, here is 0, tracking device checks IMEI and number, if at least one coincides, the command is run
- **E0** – tag E0 – command number, random number set by the server
- **00 00 00 00** – value of E0 tag = 0

- **E1** – tag E1 – command text
- **06** – value of E1 tag, text length = 6
- **73 74 61 74 75 73** – value of E1 tag, text of «status» command
- **50 22** – checksum

Byte	Value	Length, byte	Description
1	0x01	1	
2 3	L	2	Packet length
4	0x03	1	Tag
5 … 19		15	IMEI
20	0x04	1	Tag
21 22		2	Tracking device number
23	0xE0	1	Tag
24 … 27		4	Command number
28	0xE1	1	Tag
29	N	1	Reply length
30 …		N	Reply text (CP1251)

30+N			
31+N	0xEB	1	Tag
32+N	K	1	Data length
33+N … 33+N+K		K	Data
L+1 L+2		2	Checksum

Reply to command may include an additional tag with binary data (0xEB) received with reply.

Example of command in hexadecimal form in the order of receiving.

01 91 00 03 38 36 38 32 30 34 30 30 35 36 34 37 38 33 38 04 32 00 E0 00 00 00 00 E1 77 44 65 76 35 30 20 53 6F 66 74 3D 32 32 33 20 50 61 63 6B 3D 31 31 36 20 54 6D 44 74 3D 30 30 3A 32 34 3A 31 34 20 31 2E 30 31 2E 30 30 20 50 65 72 3D 31 30 20 4E 61 76 3D 32 35 35 20 4C 61 74 3D 30 2E 30 30 30 30 30 30 20 4C 6F 6E 3D 30 2E 30 30 30 30 30 30 20 53 70 64 3D 30 2E 30 20 48 44 4F 50 3D 30 2E 30 20 53 61 74 43 6E 74 3D 30 20 41 3D 30 2E 30 30 97 95

Decipherment:

- **01** – header
- **91 00** – length of 145 bytes
- **03** – tag 03 – IMEI
- **38 36 38 32 30 34 30 30 35 36 34 37 38 33 38** – value of 03 tag – «868204005647838»
- **04** – tag 04 – device number, set in settings
- **00 00** – value of 04 tag=0
- **E0** – tag E0 – command number, random number set by the server
- **00 00 00 00** – value of E0 tag = 0
- **E1** – tag E1 – command text
- **06** – value of E1 tag, text length = 6
- **E1** – tag E1 – command text
- **77** – value of E1 tag, text length = 119
- **44 65 76 35 30 20 53 6F 66 74 3D 32 32 33 20 50 61 63 6B 3D 31 31 36 20 54 6D 44 74 3D 30 30 3A 32 34 3A 31 34 20 31 2E 30 31 2E 30 30 20 50 65 72 3D 31 30 20 4E 61 76 3D 32 35 35 20 4C 61 74 3D 30 2E 30 30 30 30 30 30 20 4C 6F 6E 3D 30 2E 30 30 30 30 30 30 20 53 70 64 3D 30 2E 30 20 48 44 4F 50 3D 30 2E 30 20 53 61 74 43 6E 74 3D 30 20 41 3D 30 2E 30 30** – value of E1 tag. Reply: Dev50 Soft=223 Pack=116 TmDt=00:24:14 1.01.00 Per=10 Nav=255 Lat=0.000000 Lon=0.000000 Spd=0.0 HDOP=0.0 SatCnt=0 A=0.00
- **97 95** – checksum

Packet with Garmin FMI protocol data

Structure of packet with Garmin FMI data

Byte	Value	Description
1	0x06	Header
2 3	L	Packet length
		Garmin FMI packet
L+1 L+2		Checksum

Packet with Garmin FMI data does not require receiving confirmation form the server. When data are transmitted from the server to a navigator, the same packet structure is used. The tracking device does not send receiving confirmation. Server should configure ACK and NAK packets in accordance with description of Garmin FMI protocol, the tracking device does not configure them. In this case, the tracking device is used as GSM-modem between server and navigator.

Packet Sent Through Iridium System

Structure of packet sent through Iridium system

Byte	Value	Length, byte	Description
1	0x01	1	
2 3	L	2	Packet length
4 ...		31	Identification data of the packet tag

34			
35 ... 57		14	Tag of coordinates received via Iridium
58 ... L		L-57	Galileosky protocol data tag

Structure of identification data tag

Byte	Value	Length, byte	Description
1	0x01	1	
2	0x00	1	
3	0x1C	1	
4 … 7		4	Packet ID
8 … 22	ASCII	15	IMEI
23		1	Session status. 0, 1, 2 –transmission is correct, otherwise packet is invalid
24 …		4	Empty field

27				
28 … 31			4	Packet sending time, UTC

Structure of coordinates, received through Iridium system, tag

Byte	Bit	Value	Length, byte	Description
1		0x03	1	
2		0x00	1	
3	0x14	1		
4	8 … 2			0 – northern latitude, 1 – southern latitude
	1			0 – eastern longitude, 1 – western longitude
5			1	Latitude, degrees
6 7			2	Latitude, minutes with thousandths accuracy
8			1	Longitude, degrees
9 10			2	Longitude, minutes with thousandths accuracy
11 12			4	Radius where real coordinates of object are

13			
14			

Structure of Galileosky protocol data tag

Byte	Value	Length, byte	Description
1	0x02	1	
2 3	L	2	Data size
4 … L+3		L	Main packet or packet with compression without first 3 bytes (header and length) and checksum

Packet protocol tags

№	Tag	Description	Length, byte	Format
1	0x01	Hardware version	1	Unsigned integer
2	0x02	Firmware version	1	Unsigned integer
3	0x03	IMEI	15	ASCII string
4	0x04	Identifier of a device	2	Unsigned integer
5	0x10	Number of an archive record	2	Unsigned integer
6	0x20	Date and time	4	Unsigned integer, seconds since 1970-01-01 00:00:00 GMT
7	0x30	Coordinates in degrees, number of satellites, indication of coordinates determination correctness	9	4 lower bits: number of satellites. The next 4 bits: coordinates correctness, 0 – coordinates are correct, GLONASS/GPS module is a source,

		and source of coordinates		2 - coordinates are correct, cellular base stations are a source, other values – coordinates are incorrect. The next 4 bytes: signed integer, latitude, the value should be divided by 1000000, negative values correspond to southern latitude. Last 4 bytes: signed integer, longitude, the value should be divided by 1000000, negative values correspond to western longitude. For example, received: 07 C0 0E 32 03 B8 D7 2D 05. Coordinates correctness: 0 (coordinates are correct). Satellites number: 7 Latitude: 53.612224 Longitude: 86.890424
8	0x33	Speed in km/h and direction in degrees	4	2 lower bytes: unsigned integer, speed, the value should be divided by 10. 2 higher bytes: unsigned integer, direction, the value should be divided by 10. For example, received: 5C 00 48 08. Speed: 9.2 km/h. Direction: 212 degrees.
9	0x34	Height, m	2	Signed integer
10	0x35	One of the values: HDOP, if GLONASS/GPS module is coordinates source Error in meters, if cellular base stations are a source.	1	Unsigned integrer. In case of HDOP, the value should be divided by 10. In case of error, the value should be multiplied by 10.
11	0x40	Status of device	2	Unsigned integer, each bit corresponds to a separate unit state, see explanations

				below
12	0x41	Supply voltage, mV	2	Unsigned integer
13	0x42	Battery voltage, mV	2	Unsigned integer
14	0x43	Inside temperature, °C	1	Signed integer
15	0x44	Acceleration (this tag can only be used on tracking devices up to and including the 5.1 version)	4	10 lower bits: acceleration by X axis. Next 10 bits: acceleration by Y axis. Next 10 bits: acceleration by Z axis. 0g = 512, values less than 512 – acceleration, directed against the axis. Scale 1g=186. For example, 326 = -1g, 605 = 0,5g. Example, received: AF 21 98 15. Acceleration X: 431, Y: 520, Z: 345.
16	0x45	Status of outputs	2	Each bit, beginning with the lower one, indicates the state of a correspondent output
17	0x46	Status of inputs	2	Each bit, beginning with the lower one, indicates triggering on a correspondent input
18	0x50	Input voltage 0 Depending on settings: 1. voltage, mV, 2. number of pulses; 3. frequency,Hz.	2	Unsigned integer
19	0x51	Input voltage 1 Depending on settings: 1. voltage, mV, 2. number of pulses; 3. frequency,Hz.	2	Unsigned integer
20	0x52	Input voltage 2 Depending on settings:	2	Unsigned integer

		1. voltage, mV, 2. number of pulses; 3. frequency,Hz.		
21	0x53	Input voltage 3 Depending on settings: 1. voltage, mV, 2. number of pulses; 3. frequency,Hz.	2	Unsigned integer
22	0x58	RS232 0	2	The format depends on the port settings
23	0x59	RS232 1	2	The format depends on the port settings
24	0x70	Thermometer 0 identifier and measured temperature, °C	2	Lower byte: unsigned integer, identifier. Higher byte: signed integer, temperature. Identifier 127 with temperature -128 ºC mean a disconnection. Example, received: 01 10 Identifier: 01 Temperature: 16ºC
25	0x71	Thermometer 1 identifier and measured temperature, °C	2	Analogous to temperature sensor 1
26	0x72	Thermometer 2 identifier and measured temperature, °C	2	Analogous to temperature sensor 2
27	0x73	Thermometer 3 identifier and measured temperature, °C	2	Analogous to temperature sensor 3
28	0x74	Thermometer 4 identifier and measured temperature, °C	2	Analogous to temperature sensor 4
29	0x75	Thermometer 5 identifier and measured temperature, °C	2	Analogous to temperature sensor 5
30	0x76	Thermometer 6 identifier and measured temperature, °C	2	Analogous to temperature sensor 6

31	0x77	Thermometer 7 identifier and measured temperature, °C	2	Analogous to temperature sensor 7
32	0x90	First iButton key identification number	4	
33	0xc0	CAN-bus and CAN-LOG data (CAN_A0). Fuel used by a vehicle from the date of manufacturing, l	4	Unsigned integer, the value should be divided by 2
34	0xc1	CAN-bus and CAN-LOG data (CAN_A1). Fuel level, %; coolant temperature, ºC; Enginespeed, rpm.	4	Lower byte: fuel level, the value should be multiplied by 0.4 The second byte: coolant temperature, the value should be deducted 40. The third and fourth bytes: engine speed, values should be multiplied by 0.125. Example of data from bus in order of receiving: FA 72 50 25. Fuel level: 100%. Temperature 74ºC. Engine speed: 1194 rmp
35	0xC2	CAN-bus and CAN-LOG data (CAN_B0). Vehicle`s mileage, m.	4	Unsigned integer, the value should be multiplied by 5
36	0xC3	CAN_B1	4	
37	0xC4	CAN8BITR0 or vehicle speed from CAN-LOG, km/h	1	If speed is transmitted from CAN-LOG, the value is an unsigned integer
38	0xC5	CAN8BITR1 or the 2rd byte of prefix S CAN-LOG	1	
39	0xC6	CAN8BITR2 or the 1st byte of prefix S CAN-LOG	1	
40	0xC7	CAN8BITR3 or lower byte	1	

		of prefix S CAN-LOG		
41	0xC8	CAN8BITR4 or the 3rd byte of prefix P CAN-LOG	1	
42	0xC9	CAN8BITR5 or the 2rd byte of prefix P CAN-LOG	1	
43	0xCA	CAN8BITR6 or the 1st byte of prefix P CAN-LOG	1	
44	0xCB	CAN8BITR7 or lower byte of prefix P CAN-LOG	1	
45	0xCC	CAN8BITR8 or the first byte in the procedure for receiving of prefix WA CAN-LOG	1	
46	0xCD	CAN8BITR9 or the second byte in the procedure for receiving of prefix WA CAN-LOG	1	
47	0xCE	CAN8BITR10 or the third byte in the procedure for receiving of prefix WA CAN-LOG	1	
48	0xCF	CAN8BITR11 or the fourth byte in the procedure for receiving of prefix WA CAN-LOG	1	
49	0xD0	CAN8BITR12 or the fifth byte in the procedure for receiving of prefix WA CAN-LOG	1	
50	0xD1	CAN8BITR13 or the sixth byte in the procedure for receiving of prefix WA CAN-LOG	1	
51	0xD2	CAN8BITR14 or the seventh byte in the procedure for receiving of prefix WA CAN-LOG	1	

52	0xD3	The second iButton key identification number	4	
53	0xD4	Total mileage according to GPS/GLONASS units data, m.	4	Unsigned integer
54	0xD5	State of iButton keys, identifiers of which are set by iButton command.	1	Each bit corresponds to one key. Example, received: 05 or 00000101 in binary system. It means that the first and the third keys are connected
55	0xD6	Depending on settings: 1. CAN16BITR0 2. the 1st vehicle's axle load, kg 3. failure code OBD II	2	In case the load is on axle, the value is an unsigned integer; values should be divided by 2
56	0xD7	Depending on settings: 1. CAN16BITR1 2. the 2nd vehicle's axle load, kg 3. failure code OBD II	2	In case the load is on axle, the value is an unsigned integer; values should be divided by 2
57	0xD8	Depending on settings: 1. CAN16BITR2 2. the 3rd vehicle's axle load, kg 3. failure code OBD II	2	In case the load is on axle, the value is an unsigned integer; values should be divided by 2
58	0xD9	Depending on settings: 1. CAN16BITR3 2. the 4th vehicle's axle load, kg 3. failure code OBD II	2	In case the load is on axle, the value is an unsigned integer; values should be divided by 2
59	0xDA	Depending on settings: 1. CAN16BITR4 2. the 5th vehicle's axle load, kg	2	In case the load is on axle, the value is an unsigned integer; values should be divided by 2

		3. failure code OBD II		
60	0xDB	Depending on settings: 1. CAN32BITR0 2. total time of engine operation, h	4	In case the time of engine operation is transmitted, the value is an unsigned integer; values should be divided by 100
61	0xDC	Depending on settings: 1. CAN32BITR1 2. CAN-LOG, R prefix, fuel level, l	4	In case the fuel level is on CAN-LOG, the value is an unsigned integer; values should be divided by 10
62	0xDD	Depending on settings: 1.CAN32BITR2 2. CAN-LOG, user prefix	4	
63	0xDE	Depending on settings: 1. CAN32BITR3 2. CAN-LOG, user prefix	4	
64	0xDF	Depending on settings: 1.CAN32BITR4 2. CAN-LOG, user prefix	4	
65	0x54	Input 4 values. Depending on settings: 1. voltage, mV 2. number of pulses 3. frequency, Hz	2	Unsigned integer
66	0x55	Input 5 values. Depending on settings: 1. voltage, mV 2. number of pulses 3. frequency, Hz	2	Unsigned integer
67	0x56	Input 6 values. Depending on settings: 1. voltage, mV 2. number of pulses	2	Unsigned integer

		3. frequency, Hz		
68	0x57	Input 7 values. Depending on settings: 1. voltage, mV 2. number of pulses 3. frequency, H	2	Unsigned integer
69	0x80	Zero DS1923 sensor Identifier, measured temperature ºC and humidity %	3	Lower byte: unsigned integer, identifier. The second byte: signed integer, temperature. Higher byte: humidity, values should be multiplied by 100 and divided by 255. Example, received: 01 10 20. Identifier: 01 Temperature: 16ºC. Humidity: 12.54%
70	0x81	The 1st DS1923 sensor Identifier, measured temperature °C and humidity %.	3	Analogous to DS1923 zero sensor
71	0x82	The 2nd DS232sensor Identifier, measured temperature °C and humidity %	3	Analogous to DS1923 zero sensor
72	0x83	The 3rd DS232 sensor Identifier, measured temperature °C and humidity %	3	Analogous to DS1923 zero sensor
73	0x84	The 4th DS232 sensor Identifier, measured temperature °C and humidity %	3	Analogous to DS1923 zero sensor
74	0x85	The 5th DS232 sensor Identifier, measured	3	Analogous to DS1923 zero sensor

		temperature °C and humidity %		
75	0x86	The 6th DS232 sensor Identifier, measured temperature °C and humidity %	3	Analogous to DS1923 zero sensor
76	0x87	The 7th DS232 sensor Identifier, measured temperature °C and humidity %	3	Analogous to DS1923 zero sensor
77	0x60	RS485 [0]. Fuel level sensor with address 0	2	Unsigned integer
78	0x61	RS485 [1]. Fuel level sensor with address 1	2	Unsigned integer
79	0x62	RS485 [2]. Fuel level sensor with address 2	2	Unsigned integer
80	0x63	RS485 [3]. Fuel level sensor with address 3. Relative fuel level and temperature	3	2 lower bytes: unsigned integer, relative fuel level. Higher byte: signed integer, temperature, °C
81	0x64	RS485 [4]. Fuel level sensor with address 4. Relative fuel level and temperature	3	2 lower bytes: unsigned integer, relative fuel level. Higher byte: signed integer, temperature, °C
Tags RS485[5] - RS485[14] (0x65-0x6E) are similar to RS485[4] with numbers 82-91				
92	0x6F	RS485 [15]. Fuel level sensor with address 15. Relative fuel level and temperature.	3	2 lower bytes: unsigned integer, relative fuel level. Higher byte: signed integer, temperature, °C
93	0x88	Extended data RS232[0].	1	Signed integer

		Depending on settings: 1. Temperature from fuel level sensors connected to RS232 0, °C 2. Weight, received from weight identifier.		
94	0x89	Expanded data RS232[1]. Depending on the settings: 1. Temperature from fuel level sensors connected to Rs232[1], °C 2. Weight received from weight identifier	1	Signed integer
95	0x8A	Temperature from fuel level sensors connected to RS485 port with address 0, °C	1	Signed integer
96	0x8B	Temperature from fuel level sensors connected to RS485 port with address 1, °C	1	Signed integer
97	0x8C	Temperature from fuel level sensors connected to RS485 port with address 2, °C	1	Signed integer
98	0x78	Input 8 value. Depending on the settings, one of the options is the following: 1. voltage, mV; 2. number of pulses; frequency, Hz.	2	Unsigned integer
99	0x79	Input 9 value. Depending on the settings, one of the options is the following:	2	Unsigned integer

		1. voltage, mV; 2. number of pulses; frequency, Hz.		
100	0x7A	Input 10 value. Depending on the settings, one of the options is the following: 1. voltage, mV; 2. number of pulses; frequency, Hz.	2	Unsigned integer
101	0x7B	Input 11 value. Depending on the settings, one of the options is the following: 1. voltage, mV; 2. number of pulses; frequency, Hz.	2	Unsigned integer
102	0x7C	Input 12 value. Depending on the settings, one of the options is the following: 1. voltage, mV; 2. number of pulses; frequency, Hz.	2	Unsigned integer
103	0x7D	Input 13 value. Depending on the settings, one of the options is the following: 1. voltage, mV; 2. number of pulses; frequency, Hz.	2	Unsigned integer
104	0x21	Milliseconds	2	Unsigned integer, the number of milliseconds (0 to 999) completes the date and time value
129	0xA0	CAN8BITR15 or the eighth byte in the procedure for receiving of prefix WA CAN-LOG	1	Accessible only by a dynamic archive structure
Tags CAN8BITR16 - CAN8BITR29 (0xA1-0xAE) similar to CAN8BITR16 with numbers 130-143				

144	0xAF	CAN8BITR30	1	Accessible only by the dynamic archive structure
145	0xB0	CAN16BITR5	2	Accessible only by the dynamic archive structure
Tags CAN16BITR6 – CAN16BITR13 (0xB1-0xB8) similar to CAN16BITR5 with numbers 146-153				
154	0xB9	CAN16BITR14	2	Accessible only by the dynamic archive structure
161	0xF0	CAN32BITR5	4	Accessible only by the dynamic archive structure
Tags CAN32BITR6 – CAN32BITR13 (0xF1-0xF8) similar to CAN32BITR5 with numbers 162-169				
170	0xF9	CAN32BITR14	4	Accessible only by the dynamic archive structure
171	0x5A	REP-500 electricity meter readings	4	Unsigned integer
173	0x5B	Refrigeration unit data		See the format below
174	0x47	EcoDrive and driving style determination	4	Accessible only by the dynamic archive structure. Unsigned integer. Lower byte: acceleration. The second byte: braking. The third byte: cornering acceleration. The fourth byte: strike on bumps. All accelerations are expressed in standard units, 100 = 1g = 9,8 m/s2
175	0x5C	PressurePro tires pressure monitoring system, 34 sensors	68	Array from 34 structures per 2 bytes. Index in array corresponds to the sensor number. Data structure from sensor: Lower byte: unsigned integer, tire pressure, psi. Higher byte:

				Bit 0-2: temperature, from -40°C up to 100°C with the 20°C interval. Bit 3:1 – no connection with the sensor, 0 –sensor is connected. Bit 4: identifier of sensor battery low charge. Bit 5-7: the reason of data sending from the sensor. 000 – occassional sending. 001 – pressure decrease by 10% for PressurePro or by 12,5% for TPMS. 010 – pressure decrease by 20% for PressurePro or by 25% for TPMS. 100 – high temperature for TPMS. 101 – rapid pressure decrease for TPMS. 011 – pressure decrease by 50% for TPMS. 110 – the tire is inflated for PressurePro or high pressure for TPMS. 111 - New Magnet for PressurePro
176	0x5D	DBG-S11Ddosimeter data	3	2 lower bytes: ADER, 3V/h, unsigned integer, (xxxxxxyy yyyyyyyy – x-order, y – floating-point coefficient). Higher byte: dosimeter state. Bit 0-2: dose power and its indeterminacy value: 000 –weighted average value is typed out via 2 channels 001 –channel 1 value is typed out 010 – channel 2 value is typed out 101 – false value is typed out (device in testing mode) Bit 3 – channel 1 state: 0 – is off, 1 – is on. Bit 4: channel 1 state: 0 – OK, 1 – failure. Bit 5: channel 2 state: 0 – is off, 1 – is on. Bit 6: channel 2 state: 0 - OK, 1 - failure. Bit 7: economy mode: 0 –is off, 1 – is on.
177	0xE2	User data 0	4	

User data tags with numbers 178-183				
184	0xE9	User data 7	4	
185	0xEA	UserArray		Lower byte is array length
186		Minimum data set		
188	0x48	Expanded status of the device	2	Bit 0 is the connection state to the primary server. 1 is “connected”, 0 is “not connected”. Bit 1 is GPRS session status. 1 is “on”, 0 is “off”. Bit 2 is the sign of GSM jamming. 1 is “GSM jamming detected”, 0 is “no jamming detected”. Bit 3 is the connection state to the additional server. 1 is “connected”, 0 is “not connected”. Bit 4 is the sign of GPS/GLONASS jamming. 1 is “jamming detected”, 0 is “no jamming detected” Bit 5 is sign of connection to cable USB of device USB. 1 is “connected”, 0 is “not connected. Bit 6 – sign of SD car presence in device. 1 – present, 0 – absent.
191	0x49	Transmission channel	1	Bits 0 to 3 - transmission channel 0001 GSM 0010 WiFi 0011 BLE Bits 4 to 7 - transmission path 0001 Server 0010 Hub
192	0x11	Number of the current record in the archive	4	Unsigned integer
193	0x36	PDOP (Position Dilution of Precision). GNSS Positioning Accuracy Metric	1	Unsigned integer, the value should be divided by 10.

	0xFE	Extended tags		Length is determined by the content of the tag

Extended tags are transmitted as tag data of 0xFE.

Extended tags

№	Tag	Description	Parameter		
			Length, byte	Format	Example
1	0x0001	Tag Modbus 0	4	The result value must be divided by 100	
Modbus tags with numbers 1-31					
21	0x0021	Tag Bluetooth 0	4		
1-62 Bluetooth tags					
84	0x0060	Tag Bluetooth 63	4		
85	0x0061	Tag Modbus 32	4	The result value must be divided by 100	
Tags Modbus with numbers 33-62					
128	0x0080	Tag Modbus 63	4	The result value must be divided by 100	
129	0x0081	Cell identifier (CID)	2		
130	0x0082	Local area code (LAC)	2		
131	0x0083	Country code (MCC)	2		
132	0x0084	Operator code (MNC)	2		
133	0x0085	RSSI	1		

134	0x0086	Temperature sensor extended value tag 0	4		8600 0600801A 0600 — unsigned integer sensor ID (6), 801A — real sign value (6784), the value must be divided by 256 (26,5)
Extended temperature sensor tags numbered 1-6					
141	0x008D	Temperature sensor extended value tag 7	4		8D00 7F000080 7F00 — unsigned integer sensor ID (127), 0080 — real sign value (-32768), the value must be divided by 256 (-128)
142	0x008E	GPS satellite information tag	4		8E00 0A051EAE 0A – number of visible - 10 (1 byte, unsigned integer) 05 — number of used - 5 (1 byte, unsigned integer) 1E – SNR (signal/noise) average - 30 (1 byte, unsigned integer) 33 – SNR max - 51 (1 byte, unsigned integer)
143	0x008F	GLONASS satellite information tag	4		
144	0x0090	BAIDOU satellite information tag	4		
145	0x0091	GALILEO satellite information tag	4		
146	0x0092	Active SIM IMSI tag in hexadecimal ASCII format	15		9200 32353039393832303730323930353 1, where 32353039393832303730323930353 1 = 250998207029051
147	0x0093	Currently used SIM card slot	1		
148	0x0094	Active SIM CCID tag	20		
153	0x00A4	Modem WIFI Status	1		Tag value: 0 - Wi-Fi module disabled 1 - Turn on Wi-Fi. 2 - Turn off Wi-Fi. 3 - Set Wi-Fi to initial state. 4 - Select Wi-Fi. 5 mode - Get a list of available Wi-Fi networks. Used to scan surrounding networks.

					6 - Connect to a given Wi-Fi network (access point, AP) . 7 - Start your own access point. This state enables AP mode on the terminal, allowing other devices to connect to it. 8 - Starting the server on the AP. The server on the terminal is activated when it operates as an access point. 9 - Server session. In this mode, clients receive connections to the terminal server and process data from them. 10 - Activation of client mode (STA) when the terminal is connected to a Wi-Fi network (access point, AP) . 11 - Session in client mode. In this mode, the terminal connects to the specified servers and exchanges data with them.
154	0x00A5	Current WIFI error code	1		Tag value: 0 - No errors. Indicates no errors during Wi-Fi. 1 - operation - TCP initialization failed. Indicates a problem initializing the TCP connection. 2 - Driver initialization error. Indicates a problem when starting or initializing the driver Wi-Fi. 3 - Firmware download error. Indicates a problem when downloading or updating Wi-Fi firmware module. 4 - Error setting scan region. Indicates a problem when configuring the region to find available networks. 5 - Deinitialization error. Indicates a problem when shutting down or clearing Wi-Fi resources on the module. 6 - M2M connection error. Indicates a problem establishing a connection between M2M (Machine-to-Machine) devices. 7 - Access Point (AP) connection failure. Indicates a problem when trying to connect to a Wi-Fi network. 8 - Access point startup error. Indicates a problem when trying to start the device in access point mode. 9 - Error getting RSSI value (signal strength). Indicates a problem while trying to measure the Wi-Fi signal level. 10 - Access point disconnect error. Indicates a problem when trying to disable access point mode. 11 - Client Shutdown Error (STA). Indicates a problem when trying to disable client mode Wi-Fi. 12 - WLAN break time error. Indicates a problem with the connection break time interval Wi-Fi. 13 - Error getting firmware information. Indicates a problem when trying to get information about the current firmware

					version. 14 - Error getting MAC address. Indicates a problem when trying to get the Wi-Fi MAC address of the module.
155	0x00A6	GSM modem status	1		Tag Value: 0 - Initialized. Indicates that the system has been successfully initialized and is running normally. 1 - Powered up. Indicates that the device is powered on and running. 2 - Session restart required. Indicates that the GPRS session will be restarted. 3 - Module restart required. Indicates whether the device module will be restarted. 4 - Power is off. Indicates that the device is powered off.
156	0x00A7	Network registration status	1		Tag value: 0 - Not registered, the device does not look for an operator to register. Indicates that the device is not registered on the network and is not currently looking for available operators to connect. 1 - Registered, home network. Indicates that the device has successfully registered with its home network. 2 - Not registered, but the device is currently looking for a new operator to register. Indicates that the device is not registered but is actively looking for available networks to connect. 3 - Registration denied. Indicates that an attempt to register with the network has been rejected. 4 - Unknown (e.g. out of GERAN/UTRAN coverage). Indicates that the registration status is unknown, possibly due to lack of network coverage. 5 - Registered, roaming. Indicates that the device is registered on the network but roaming (outside the home network). 6 - Registered for "SMS only," home network. Indicates that the device is registered on its home network, but only for sending and receiving SMS. 7 - Registered for SMS only, roaming. Indicates that the device is registered to send and receive SMS while roaming. 8 - Counter of registration status types. Specifies the number of different types of registration statuss. 255 - Undefined. Indicates that the registration status is uncertain or unknown.
157	0x00A8	GPRS status	1		Established GPRS Session Feature: 1 - Session Active 0 - Session Inactive

158	0x00A9	Amount of free RAM	4	Unsigned integer. Value in bytes	
160	0x00AB	Status of records in archive	12	Byte 0-3: total number of points (unsigned integer) Bytes 4-7: number of points sent to primary server (unsigned integer) Bytes 8-11: number of points sent to secondary server (unsigned integer)	2E3F0000 3E020000 DD040000, where 00003F2E is the unsigned integer total number of points (16174) 00003E02 is the unsigned integer number of points sent to the primary server (15874) 000004DD is the unsigned integer number of points sent to the secondary server (1245)
161	0x00AC	Number of the last record in the archive	4	Unsigned integer	
163	0x00AD	MAC address WiFi	6	MAC address in HEX format	0080C25E265A
162	0x00AE	MAC address BLE	6	MAC address in HEX format	80EACA004F3A
164	0x00AF	self-troubleshooting	14	Data Structure: Bytes 0-7: Last Reset Date and Time (UNIX time) Bytes 8-9: Device Reboot Reason Bytes 10-13: Number of reboots due to 8-9 bytes	1700 - System error in autoinformer operation 100 - System error in GNSS module operation 0 - System error during GPRS 1200 - operation - System error in power supply circuit 400 - System error when working with SD card or eMMC memory 500 - Task system error I2C 503 - Accelerometer System Error 600 - 1-Wire 1300 Interface System Error - System Task Error Outs 1301 - Output State Control Errors 1400 - System error in processing IN 1401 input states - System error in system power control (Battery, USB, external voltage) 1602 - Audio system error (autoinformer, voice communication on terminals with ublox 3G) 300 - System error when writing to memory 301 - System error when reading from memory 1900-1908 - Processor errors 8982DF6700000000 F701 09000000, where 0000000067DF8289 - date 03.23.2025 03:39:53 (1742701193 sec) 01F7 - unsigned integer reason for rebooting the device (503)

					00000009 - unsigned integer number of reboots of the device (9)
165	0x00B0	Total mean SNR	1	Unsigned integer	If value: > 50 - level excellent from 30 to 50 - level good from 10 to 30 - level satisfactory < 10 - level poor
166	0x00B1	SD card status	1	Unsigned integer	Tag value: 0 - Initialize and power up. 1 - Initialize MSD 2 - mode - MSD 3 - mode - Mount FS 4 - Monitor terminal, memory card and file system 5 - Deinitialize SD card
167	0x00B2	SD card errors	1	Unsigned integer	Tag Value: 0 - No Errors 1 - SD Card Not Found or No External Feed 2 - Failed to Mark File as Shipped 3 - Failed to Get Main Data Package 4 - Failed to Mark Record 5 - Failed to Write
168	0x00B3	Collector Archive Status	12	Byte 0-3: Total packets (unsigned integer) Bytes 4-7: Number of packets sent to primary server (unsigned integer) Bytes 8-11: Reserve	2E3F0000 3E020000 DD040000 where 00003F2E is the unsigned integer total number of packets (16174) 00003E02 is the unsigned integer number of packets sent to the primary server (15874)
169	0x00B4	Client MAC address 1	6	MAC address in HEX format	0080C25E4F3A
170	0x00B5	Client MAC address 2	6	MAC address in HEX format	
171	0x00B6	Client MAC address 3	6	MAC address in HEX format	
217	0x00D9	TMPS wheel tag 0	3		Structure of the data from the sensor: **Byte 0**: unsigned integer, tyre pressure, psi **Byte 1**: signed integer, temperature, °C

					Byte 2: Bit 0: 1 - no communication with sensor. 0 - sensor is communicating Bit 1: sign of low sensor battery or sensor error Bit 2-4: the reason for sending data from the sensor 000 - periodic sending. 001 - 10% pressure loss for PressurePro or 12.5% TPMS. 010 - 20% pressure loss for PressurePro or 25% pressure loss for TPMS. 100 - high temperature for TPMS. 101 - rapid pressure drop for TPMS. 011 - 50% loss of pressure for TPMS. 110 - tyre re-inflated for PressurePro or high pressure for TPMS. 111 - New Magnet for PressurePro
	TMPS wheel tags 217 to 250				
250	0x00FA	TMPS wheel tag 33	3		
252	0x00FC	Reason for recording an archive point	1		Tag values: 1 - Periodic recording by device settings 2 - iButton key events 3 - Data from DataCOLD500 received 4 - Data from EuroScan received 5 - Data from ThermoKing received 8 - Device status changed 9 - User record from pawn algorithm or script 10 - Inputs event 11 - Distance specified by the user in the settings was covered 12 - Alarm by signalling settings was triggered 13 - Emergency signal

253	0x00FD	iButton64 tag	8		
254	0x00FE	iButton64 2 tag	8		
10020	0x2724	Engine Coolant Pressure 1 (Extended Range), kPa	size depends on the tag content		
SPN tags 10021 to 32768					
32769	0x8001	Brake Wear Life Remaining, Trailer Axle #8, Left Wheel, %	size depends on the tag content		

A complete list of SPN tags/parameters for J1939 protocol is available here.

A complete list of SPN tags/parameters for ISOBUS protocol is available here.

The tag data is transmitted in the following format:

Number of bytes in the sensor value field is determined in this table

Packet example:

01 2300 10 0000 FE 1D00 2427 02 00 0000 01 0000 2627 01 0000000000 2B27 01 00 0000000000000000 4BCE

01 - main packet header

2300 - packet size

10 - "point number" tag header

0000 - tag value

FE - extended tags

1D00- data size in extended tags

2427 - tag header

02 - number of sensors

00 - sensor 1 address

0000 - sensor 1 value

01 - sensor 2 address

0000 - sensor 2 value

2627 - tag header

01 - number of sensors

00 - sensor 1 address

00000000 - sensor 1 value

2B27 - tag header

01 - number of sensors

00 - sensor 1 address

0000000000000000 - sensor 1 value

4BCE - crc

Device status field explanation

Bit number	Field explanation
0	0 – vibration level corresponds to parking; 1 – to driving (set by AccSens command)
1	0 – incline angle does not exceed the allowable one, 1 – incline level exceeds the allowable one
2	0 – none of the trusted iButton keys are connected, 1- one of the recorded to the SD-card iButton keys is connected
3	0 – there is a SIM card, 1 – GSM/3G-unit can’t determine the SIM-card
4	0 – tracking device is outside the geofence, 1 - tracking device is inside the geofence
5	0 – voltage of internal source is normal; 1 – lower than 3.7 V
6	0 – GPS aerial is connected; 1 – disconnected
7	0 – voltage of internal Tracking device bus supply is normal, 1 – declined from normal
8	0 – external supply voltage is normal, 1 - declined from normal (set by powincfg command)
9	0 – vehicle is stopped; 1 – vehicle is started (set by mhours command)
10	0 – vibration level corresponds to the normal movement, 1 – vibration level corresponds to a strike
11	For devices with built-in GPS module (without GLONASS support): 0 – coordinates of built-in module are used; 1 – coordinates of external module are used (for example, GLONASS adaptor). For devices with built-in GLONASS/GPS module: 0 – coordinates of external module are used (for example Trimble guidance system); 1 – coordinates of built-in module are used
12	Signal quality, range: [0-3]. The less value, the worse communication.

13	
14	0 – signaling mode is off; 1 – on.
15	0 – no alarm; 1 – alarm activated.