FMB920 Protocols

V0.09

Contents

1. FMB920 DATA PROTOCOL

1.1 *AVL data packet*

Below table represents AVL data packet structure.

4 zeroes	Data field length	Codec ID	Number of Data 1	AVL Data	Number of Data 2	CRC-16
4 Bytes	*4 Bytes*	*1 Byte*	*1 Byte*	*30- 147 Bytes*	*1 Byte*	*4 bytes*

Number of data – number of encoded data (number of records).
In FM920 codec ID is constant 08.
Data field length is the length of bytes [codec id, number of data 2].
Number of data 1 should always be equal to number of data 2 byte.
CRC-16 is 4 bytes, but first two are zeroes and last two are CRC-16 calculated for [codec id, number of data 2]
Minimum AVL packet size is 45 bytes (all IO elements disabled).
Maximum AVL packet size for one record is 783 bytes

1.2 *AVL Data*

Timestamp	Priority	GPS Element	IO Element
8 Bytes	*1 Byte*	*15 Bytes*	*6-123*

Timestamp – difference, in milliseconds, between the current time and midnight, January 1, 1970 UTC.

1.3 *Priority*

0	Low
1	High
2	Panic

1.4 *GPS Element*

Longitude	Latitude	Altitude	Angle	Satellites	Speed
4 Bytes	*4 Bytes*	*2 Bytes*	*2 Bytes*	*1 Byte*	*2 Bytes*

X	Longitude[1]
Y	Latitude[1]
Altitude	In meters above sea level[1]
Angle	In degrees, 0 is north, increasing clock-wise [1]
Satellites	Number of visible satellites[1]
Speed	Speed in km/h. 0x0000 if GPS data is invalid[1]

Longitude and latitude are integer values built from degrees, minutes, seconds and milliseconds by formula.

$$\left(d+\frac{m}{60}+\frac{s}{3600}+\frac{ms}{3600000}\right)*p$$

d	Degrees
m	Minutes
s	Seconds
ms	Milliseconds
p	Precision (10000000)

If longitude is in west or latitude in south, multiply result by –1. To determine if the coordinate is negative, convert it to binary format and check the very first bit. If it is 0, coordinate is positive, if it is 1, coordinate is negative.

Example:

Received value: 20 9c ca 80
Converted to BIN: 00100000 10011100 11001010 10000000 first bit is 0, which means coordinate is positive
Convered to DEC: 547146368
For more information see two‘s compliment arithmetics.

1.5 IO element

Event IO ID	N of **Total IO**	N1 of **One Byte** IO	**1'st** IO ID	**1'st** IO Value	…	**N1'th** IO ID	**N1'th** IO Value	N2 of **Two Bytes** IO	**1'st** IO ID	**1'st** IO Value	…	**N2'th** IO ID	**N2'th** IO Value	N4 of **Four Bytes** IO	**1'st** IO ID	**1'st** IO Value	…	**N4'th** IO ID	**N4'th** IO Value	N8 of **Eight Bytes** IO	**1'st** IO ID	**1'st** IO Value	…	**N8'th** IO ID	**N8'th** IO Value
1 Byte	*1 Byte*	*1 Byte*	*1 Byte*	*1 Byte*		*1 Byte*	*1 Byte*	*1 Byte*	*1 Byte*	*2 Bytes*		*1 Byte*	*2 Bytes*	*1 Byte*	*1 Byte*	*4 Bytes*		*1 Byte*	*4 Bytes*	*1 Byte*	*1 Byte*	*8 Bytes*		*1 Byte*	*8 Bytes*

Event IO ID – if data is acquired on event – this field defines which IO property has changed and generated an event. If data cause is not event – the value is 0.

[1] If record is without valid coordinates – (there were no GPS fix in the moment of data acquisition) – Longitude, Latitude and Altitude values are last valid fix, and Angle, Satellites and Speed are 0.

N	total number of properties coming with record (N=N1+N2+N4+N8)
N1	number of properties, which length is 1 byte
N2	number of properties, which length is 2 bytes
N4	number of properties, which length is 4 bytes
N8	number of properties, which length is 8 bytes

Permanent I/O elements (are always sent (with every record) to server if enabled)			
Property ID in AVL packet	Property Name	Bytes	Description
239	Ignition	1	Logic: 0 / 1 * Depends on Ignition source
240	Movement	1	Logic: 0 / 1 * Depends on Movement source
80	Data Mode	1	Value in scale 0 – 5 0 – Home On Stop 1 – Home On Moving 2 – Roaming On Stop 3 – Roaming On Moving 4- Unknown On Stop 5 – Unknown On Moving
21	GSM Signal Strength	1	Value in scale 1 – 5
200	Sleep Mode	1	0 – No Sleep; 1 – GPS Sleep; 2 – Deep Sleep; 3- Online Sleep
69	GNSS Status	1	0 - OFF 1 - ON with fix 2 - ON without fix 3 - In sleep state
181	PDOP	2	Probability * 10; 0-500
182	HDOP	2	Probability * 10; 0-500
66	Ext Voltage	2	Voltage: mV, 0 – 30 V
24	Speed	2	Value in km/h, 0 – xxx km/h
205	GSM Cell ID	2	GSM base station ID
206	GSM Area Code	2	Location Area code (LAC), it depends on GSM operator. It provides unique number which assigned to a set of base GSM stations. Max value: 65536
67	Battery Voltage	2	Voltage: mV
68	Battery Current	2	Current: mA
241	GSM Operator	4	Currently used GSM Operator code
199	Trip Odometer	4	Trip Odometer Value in meters
16	Total Odometer	4	Total Odometer Value in meters
1	Din 1	1	Logic: 0 / 1
9	Ain 1	2	Voltage: mV, 0 – 30 V
179	DOUT 1	1	Logic: 0 / 1
12	Fuel Used GPS	4	Fuel Used in mili Liters
13	Average Fuel Use	2	Average Fuel use in (Litersx100) /100km
17	Accelerometer X axis	2	X axis: value mG range [-8000; 8000]
18	Accelerometer Y axis	2	Y axis: value mG range [-8000; 8000]
19	Accelerometer Z axis	2	Z axis: value mG range [-8000; 8000]
*11	SIM ICCID number part 1	8	Value of SIM ICCID, MSB (Example Below)
*14	SIM ICCID number part 2	8	Value of SIM ICCID, LSB (Example Below)

Permanent I/O elements (are always sent (with every record) to server if enabled)			
Property ID in AVL packet	Property Name	Bytes	Description
10	SD Status	1	0 – not present, 1 – present

There are 8 IO elements of 1 byte size.

Also 13 IO elements of 2 byte size.

Also 4 IO elements of 4 byte size.

And 0 IO elements of 8 byte size.

```
*ICCID Full Value Calculation, Example
1)    Calculate ID:14 lenght as string
2)    If  lenght < 10, then add_zeros = 10 - length
3)    Else no zeros must be added
4)    Concat strings to get final value. Final value = String(ID 11) +
   String(add_zeros) + String(ID 14).
```

ID:11 Len as string	ID:14 Len as string	Full Value	Full Value Len
9	9	String(ID 11) + „0“ + String(ID 14)	19
9	10	String(ID 11) + String(ID 14)	19
10	10	String(ID 11) + String(ID 14)	20
9	11	String(ID 11) + String(ID 14)	20
11	8	String(ID 11) + „00“ + String(ID 14)	21
11	10	String(ID 11) + String(ID 14)	21
12	10	String(ID 11) + String(ID 14)	22
12	9	String(ID 11) + „0“ + String(ID 14)	22

Eventual I/O elements (Send if corresponding event had happen)			
Property ID in AVL packet	Property Name	Bytes	Description
155	Geofencing Zone 1 Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;

Eventual I/O elements (Send if corresponding event had happen)			
Property ID in AVL packet	Property Name	Bytes	Description
156	Geofencing Zone 2 Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;
157	Geofencing Zone 3 Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;
158	Geofencing Zone 4 Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;
159	Geofencing Zone 5 Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;
175	AutoGeofence Event	1	Logic: 0 / 1 0 – Exit Event; 1 – Enter Event;
250	Trip Event	1	Logic: 0 / 1 0 – Trip Ended; 1 – Trip Started;
255	Overspeeding Event	1	Value km/h that generated event
251	Idling Event	1	Logic: 0 / 1 0- Idling ended event; 1 – Idling started event;
253	Green Driving Type	1	Possible Values: [1/2/3] 1 – Acceleration 2 – Braking 3 – Cornering
254	Green Driving Value	1	Depending on eco driving type: if harsh acceleration, braking and cornering – g*10
246	Towing Detection Event	1	1 – Send Towing detected
252	Unplug Event	1	1 – Send when unplug event happens
247	Crash Detection	1	1 – Crash Detected 2 – Crash Trace Record, (begins 5 sec before crash, and ends 5 sec after crash */
249	Jamming Detection	1	1 – Jamming Detected 0 – Jamming Ended

Permanent I/O elements (Send if ask to get with OBDII dongle)			
Property ID in AVL packet	Property Name	Bytes	Description

Permanent I/O elements (Send if ask to get with OBDII dongle)			
Property ID in AVL packet	Property Name	Bytes	Description
30	„Number of DTC“	1	
31	„Calculated engine load value“	1	%
32	„Engine coolant temperature“	1	C
33	„Short term fuel trim 1“	1	%
34	„Fuel pressure“	2	kPa
35	„Intake manifold absolute pressure“	1	kPa
36	„Engine RPM“	2	rpm
37	„Vehicle speed“	1	km/h
38	„Timing advance“	1	O
39	„Intake air temperature“	1	C
40	„MAF air flow rate“	2	g/sec, *0.01
41	„Throttle position“	1	%
42	„Run time since engine start“	2	s
43	„Distance traveled MIL on“	2	Km
44	„Relative fuel rail pressure“	2	kPa, *0.1
45	„Direct fuel rail pressure“	2	kPa, *0.1
46	„Commanded EGR“	1	%
47	„EGR error“	1	%
48	„Fuel level“	1	%
49	„Distance traveled since codes cleared“	2	Km
50	„Barometric pressure“	1	kPa
51	„Control module voltage“	2	mV
52	„Absolute load value“	2	%
53	„Ambient air temperature“	1	C
54	Time run with MIL on	2	Min
55	„Time since trouble codes cleared“	2	Min
56	„Absolute fuel rail pressure“	2	kPa, *10
57	„Hybrid battery pack remaining life“	1	%
58	„Engine oil temperature“	1	C
59	„Fuel injection timing“	2	O, *0.01
60	„Engine fuel rate“	2	L/h, *100

To receive CAN data, send if ask to get with OBDII dongle. FMB9 module CAN data is not reading.

1.6 Example

Received data:

```
000000000000008c08010000013feb55ff74000f0ea850209a6900009400001200000001e0
9010002000300040016014703f0001504c8000c0900730a00460b00501300464306d74400
00b5000bb60007422e9f180000cd0386ce000107c700000000f10000601a4600000134480
0000bb84900000bb84a00000bb84c00000000024e0000000000000000cf00000000000000
000100003fca
```

In total 152 Bytes.

00000000 **4 zeroes**, 4 bytes

0000008c **data length**, 4 bytes

08 – **Codec ID**

0- **Number of Data** (1 record)

1'st record data

0000013feb55ff74 – **Timestamp** in milliseconds (1374042849140)
GMT: Wed, 17 Jul 2013 06:34:09 GMT
00 – **Priority**

GPS Element

0f0ea850 – Longitude 252618832 = 25,2618832° N
209a6900 – Latitude 546990336 = 54,6990336 ° E
0094 – Altitude 148 meters
0 – Angle 214°
12 – 12 Visible sattelites
0 – 0 km/h speed

IO Element

00 – IO element ID of Event generated (in this case when 00 – data generated not on event)
1e – 30 IO elements in record (total)
09 – 9 IO elements, which length is 1 Byte
0 – IO element ID = 01
0 – IO element's value = 0
02 – IO element ID = 02
0 – IO element's value = 0
03 – IO element ID = 03
0 – IO element's value = 0
04 – IO element ID = 04
0 – IO element's value = 0
16 – IO element ID = 22 (dec)
0 – IO element's value = 1
47 – IO element ID = 71 (dec)
03 – IO element's value = 3

```
  F0      - IO element ID = 240 (dec)
  0       - IO element’s value = 0
  15      - IO element ID = 21 (dec)
  04      - IO element’s value = 0
  C8      - IO element ID = 200 (dec)
  0       - IO element’s value = 0

0C    - 12 IO elements, which value length is 2 Bytes
  09      - IO element ID = 9 (dec)
  0073    - IO element’s value
  0a      - IO element ID = 10 (dec)
  0046    - IO element’s value
  0b      - IO element ID = 11 (dec)
  0050    - IO element’s value
  13      - IO element ID = 19 (dec)
  0046    - IO element’s value
  43      - IO element ID = 67 (dec)
  06d7    - IO element’s value
  44      - IO element ID = 68 (dec)
  0       - IO element’s value
  B5      - IO element ID = 181 (dec)
  000b    - IO element’s value
  B6      - IO element ID = 182 (dec)
  0007    - IO element’s value
  42      - IO element ID = 66 (dec)
  2e9f    - IO element’s value
  18      - IO element ID = 24 (dec)
  0       - IO element’s value
  cd      - IO element ID = 205 (dec)
  0386    - IO element’s value
  CE      - IO element ID = 206 (dec)
  0       - IO element’s value

07 - 7 IO elements, which value length is 4 Bytes

  C7          - IO element ID = 199 (dec)
  0       - IO element’s value
  f1          - IO element ID = 241 (dec)
  0000601a    - IO element’s value
  46          - IO element ID = 70 (dec)
  00000134    - IO element’s value
  48          - IO element ID = 72 (dec)
  00000bb8    - IO element’s value
  49          - IO element ID = 73 (dec)
  00000bb8    - IO element’s value
  4a          - IO element ID = 74 (dec)
  00000bb8    - IO element’s value
  4c          - IO element ID = 76 (dec)
  0       - IO element’s value
```

```
          02 - 2 IO elements, which value length is 8 Bytes
             4e                   - IO element ID = 78 (dec)
             0        - IO element's value
             cf                   - IO element ID = 207 (dec)
             0        - IO element's value

  0    –Number of Data (1 record)
  00003fca      - CRC-16, 4 Bytes (first 2 are always zeroes)
```

2. SENDING DATA OVER TCP/IP

First when module connects to server, module sends its IMEI. First comes short identifying number of bytes written and then goes IMEI as text (bytes).

For example IMEI 356307042441013 would be sent as 000f333536333037303432343431303133

First two bytes denote IMEI length. In this case 000F means, that imei is 15 bytes long.

After receiving IMEI, server should determine if it would accept data from this module. If yes server will reply to module 01 if not 00. Note that confirmation should be sent as binary packet. I.e. 1 byte 0x01 or 0x00.

Then module starts to send first AVL data packet. After server receives packet and parses it, server must report to module number of data received as integer (four bytes).

If sent data number and reported by server doesn't match module resends sent data.

Example:
Module connects to server and sends IMEI:
000f333536333037303432343431303133
Server accepts the module:
01
Module sends data packet:

AVL data packet header	*AVL data array*	*CRC*
Four zero bytes, 'AVL data array' length – 254	CodecId – 08, NumberOfData – 2. (Encoded using continuous bit stream. Last byte padded to align to byte boundary)	CRC of 'AVL data array'
00000000000000FE	0802...(data elements)...02	00008612

Server acknowledges data reception (2 data elements):
00000002

3. SENDING DATA OVER UDP/IP

3.1 *UDP channel protocol*

UDP channel is a transport layer protocol above UDP/IP to add reliability to plain UDP/IP using acknowledgment packets. The packet structure is as follows:

UDP datagram			
UDP channel packet x N	Packet length	2 bytes	Packet length (excluding this field) in big endian byte order
	Packet Id	2 bytes	Packet id unique for this channel
	Packet Type	1 byte	Type of this packet
	Packet payload	m bytes	Data payload

Packet Type	
0	Data packet requiring acknowledgment
1	Data packet NOT requiring acknowledgment
2	Acknowledgment packet

Acknowledgment packet should have the same *packet id* as acknowledged data packet and empty data payload. Acknowledgement should be sent in binary format.

Acknowledgment packet		
Packet length	2 bytes	0x0003
Packet id	2 bytes	same as in acknowledged packet
Packet type	1 byte	0x02

3.2 *Sending AVL data using UDP channel*

AVL data are sent encapsulated in UDP channel packets (*Data payload* field).

AVL data encapsulated in UDP channel packet		
AVL packet id (1 byte)	Module IMEI	AVL data array

AVL packet id (1 byte) – id identifying this AVL packet
Module IMEI – IMEI of a sending module encoded the same as with TCP

AVL data array – array of encoded AVL data

Server response to AVL data packet	
AVL packet id (1 byte)	Number of accepted AVL elements (1 byte)

AVL packet id (1 byte) – id of received AVL data packet

Number of AVL data elements accepted (1 byte) – number of AVL data array entries from the beginning of array, which were accepted by the server.

Scenario:

Module sends UDP channel packet with encapsulated AVL data packet (*Packet* type=1 or 0). If packet type is 0, server should respond with valid UDP channel acknowledgment packet. Since server should respond to the AVL data packet, UDP channel acknowledgment is not necessary in this scenario, so *Packet type=1* is recommended.

Server sends UDP channel packet with encapsulated response (*Packet type=1* – this packet should not require acknowledgment)

Module validates *AVL packet id* and *Number of accepted AVL elements*. If server response with valid *AVL packet id* is not received within configured timeout, module can retry sending.

Example:

Module sends the data:

UDP channel header	***AVL packet header***	***AVL data array***
Len – 253, Id – 0xCAFE, Packet type – 01 (without ACK)	AVL packet id – 0xDD, IMEI – 1234567890123456	CodecId – 08, NumberOfData – 2. (Encoded using continuous bit stream)
00FDCAFE01	DD000F313233343536373839303132333435	0802…(data elements)…02

Server must respond with acknowledgment:

UDP channel header	***AVL packet acknowledgment***
Len – 5, Id – 0xABCD, Packet type – 01 (without ACK)	AVL packet id – 0xDD, NumberOfAcceptedData – 2
0005ABCD01	DD02

Another example, with all IO id’s enabled

```
Server received data:
00a1cafe011b000f333536333037303432343431303133080100000013febdd19c8000f0e9
ff0209a7180006900001200000001e09010002000300040016014703f0001504c8000c0900
910a00440b004d130044431555440000b5000bb60005422e9b180000cd0386ce000107c70
0000000f10000601a460000013c4800000bb84900000bb84a00000bb84c00000000024e00
0000000000000cf000000000000000001
```

Data length: 00a1 or 161 Bytes (not counting the first 2 data length bytes)

Packet identification:	0xCAFE 2 bytes
Packet type:	01
Packet id:	1b
Imei length:	000f
Actual imei:	333536333037303432343431303133
Codec id:	08
Number of data:	01
Timestamp:	0000013febdd19c8
Priority:	00
GPS data:	0f0e9ff0209a718000690000120000

UDP protocol is the same as TCP except message header is 7 bytes, which consist of: data length, packet identification, packet type and packet id.

Then goes imei length and imei itself.

And after that goes AVL data.

And at the very end number of data byte. There is no CRC in UDP.

4. SENDING DATA USING SMS

AVL data or events can be sent encapsulated in binary SMS. TP-DCS field of these SMS should indicate that message contains 8-bit data (for example: TP-DCS can be 0x04).

SM data (TP-UD)	
AVL data array	*IMEI:* 8 bytes

AVL data array – array of encoded AVL data
IMEI – IMEI of sending module encoded as a big endian 8-byte long number.

5. SMS EVENTS

When Configured to generate SMS event user will get this SMS upon event

<Year/Month/Day> <Hour:Minute:Second> Lon:<longitude> **Lat**:<latitude> **Q:**<HDOP> **<SMS Text>** **Val**:<Event Value>

Example:
2016/04/11 12:00:00 Lon:51.12258 Lat: 25.7461 Q:0.6 Digital Input 1 Val:1

6. CHANGE LOG

Nr.	Date	New version number	Comments
1	2016.10.02	0.0.1	First release
2	2016.11.15	0.0.3	Minor changes
3	2017.01.24	0.0.4	OBD AVL ID
4	2017.03.30	0.0.5	Added ICCID and SD status.
5	2017.04.24	0.0.6	GPS AVG Fuel Use in 100km. Multiplier (x100) info added CCID ID is put to two IO elements (AVL ID:11 and AVL ID:14), parsing instructios added
6	2017.06.16	0.0.7	Updated IO GNSS status values
7	2017.07.03	0.0.8	Description added: ICCID Full Value Calculation
8	2017.07.25	0.0.9	Updated OBD fuel rate param.