FM Pro, FM Eco & FM Tco Protocols version 1.44

Table of Contents

1 Records

1.1 *Record structure*

All records have defined structure which can vary in length. Record consists of 2 parts: header with fixed length (23 bytes) and body with varying length (4-103 bytes). Maximum record size is 126 bytes.

Header [23 B]	Body [4-103 B]

All data are in hex format.

1.2 *Record header*

All headers have the same parameters' fields. These fields are showned below.

Time stamp [4B]	Time stamp extension [1B]	Priority [1B]	Longitude [4B]	Latitude [4B]	Altitude [2B]	Angle [2B]	Satellites [1B]	Speed [2B]	HDOP [1B]	Event ID which generates record [1B]

1.2.1 Time stamp

Time stamp – difference, in seconds, between the current time and midnight, January 1, 1970 UTC (Unix time stamp: http://www.unixtimestamp.com). Parameter length – 4 bytes.

1.2.2 Time stamp extension

Time stamp extension – an extra byte to separate records with same time stamp. If some records have same time stamp when time stamp extension will increase starting with zero <0x00>. If there are no records with same time stamp parameter will always be zero. Parameter length – 1 byte.

See more: *1.5 Extended records* (page 6).

1.2.3 Priority

Priority can be low or high. It depends on configuration of event which triggered record. Parameter length – 1 byte.

Priority	Explanation
High	Event is configured as high priority. Initiates data sending instantly. Value <0x01>
Low	Event is configured as low priority. Value <0x00>

1.2.4 GPS element – Longitude

It is GPS element. Together with latitude and altitude it reveals the position of an object. Parameter length – 4 bytes. Longitude is an integer number which calculated by formula:

$$long = (d + \frac{m}{60} + \frac{s}{3600} + \frac{ms}{3600000}) \times p \qquad (1).$$

d – Degrees, m – Minutes, s – Seconds, ms – Milliseconds, p – Precision = 10000000. Result is multiplied by –1 if longitude is in west.

1.2.5 GPS element – Latitude

It is GPS element. Together with longitude and altitude it reveals the position of an object. Parameter length – 4 bytes. Parameter length – 4 bytes. Latitude is an integer number which calculated by formula:

$$lat = (d + \frac{m}{60} + \frac{s}{3600} + \frac{ms}{3600000}) \times p \qquad (2).$$

d – Degrees, m – Minutes, s – Seconds, ms – Milliseconds, p – Precision = 10000000. Result is multiplied by –1 if latitude is in south.

1.2.6 GPS element – Altitude

Parameter is in meters above sea level. Value is multiplied by 10. Parameter length – 2 bytes.

1.2.7 GPS element – Angle

Parameter is in degrees. Value zero <0x0000> is north, increasing clock-wise. Value is multiplied by 100. Parameter length – 2 bytes.

1.2.8 GPS element – Satellites

It is a number of visible GPS or GLONASS satellites (depends on device configuration). Parameter length – 1 byte.

1.2.9 GPS element – Speed

Object's current speed in km/h. Parameter length – 2 bytes.

1.2.10 GPS element – HDOP

Horizontal Dilute Of Precision parameter is a factor in determining the relative accuracy of a horizontal position. Value is multiplied by 10. The smaller the DOP number, the better the geometry. (more: http://en.wikipedia.org/wiki/Dilution_of_precision_(GPS)). Value is multiplied by 10.parameter length – 1 byte.

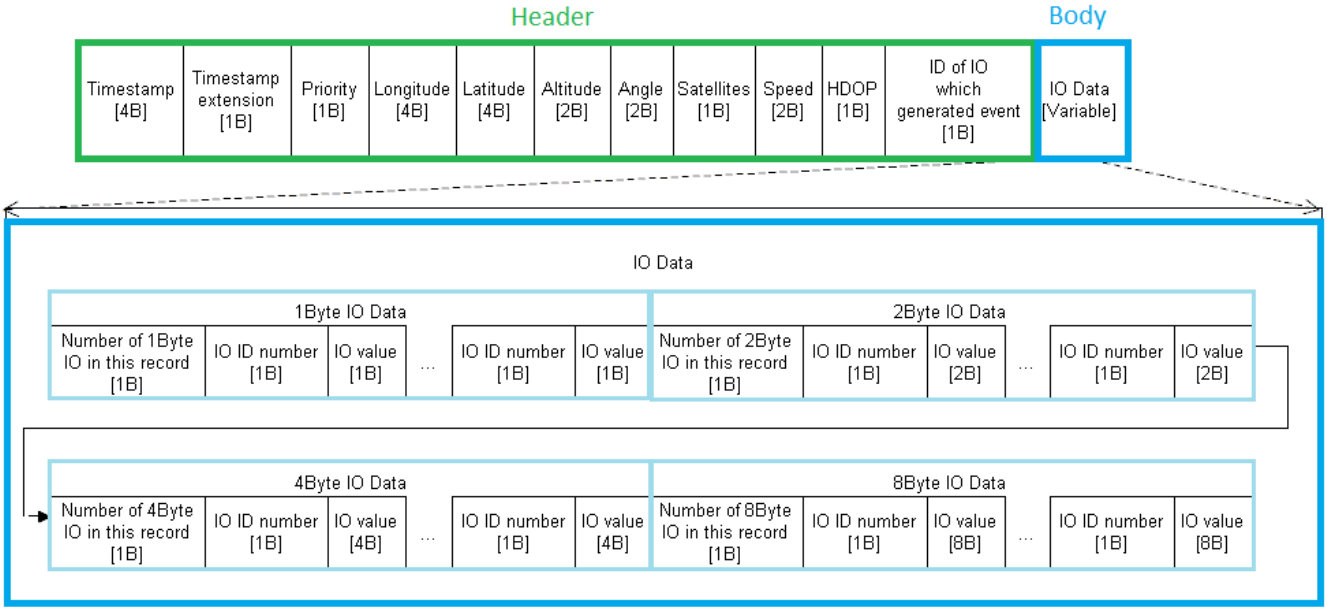
1.2.11 Event ID which generates record

Parameter indicates why the record was created. Value is event ID number. Parameter length – 1 byte.

If FM device at the time when record was generated did not have valid coordinates (there were no GPS/GLONASS fix in the moment of data acquisition) then parameters *Longitude*, *Latitude, Altitude, Angle* values would be last valid fix. *HDOP*, *Satellites* and *Speed* would be cleared to 0.

1.3 *Record body*

Record body can vary in length. Length depends on configuration of the FM device. Body is devided into segmens of different length parameters (1, 2, 4 and 8 bytes long). Every segment holds the number of parameters with same size and ID numbers. Record structure is shown below.

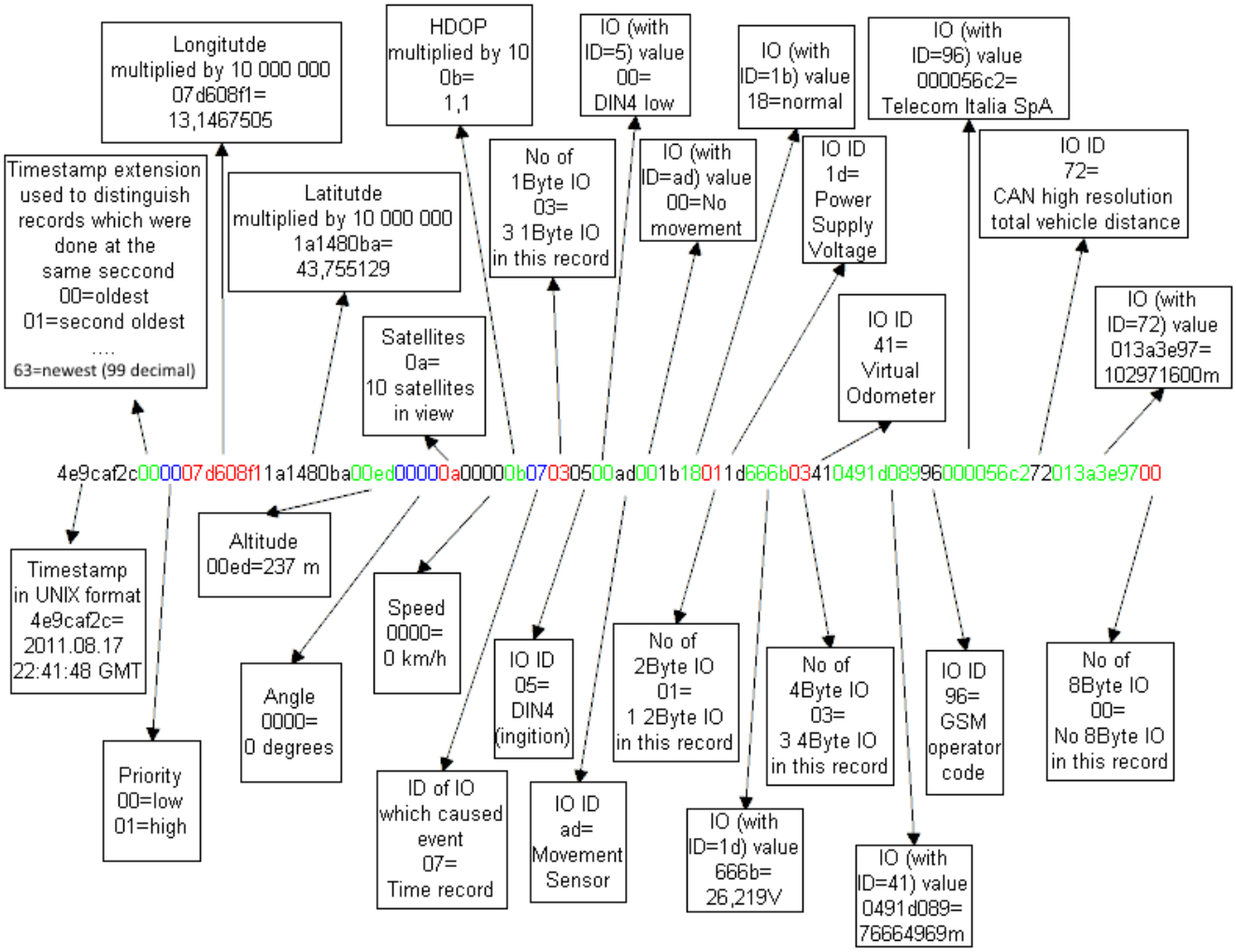


When data collection without GPS fix is enabled and no GPS fix acquired, all GPS realated fields which can be positive or negative will have <0x80..00> value. Fields which can only be positive will have <0xFF..FF> value.

GPS_fix	Longitude	Latitude	Altitude	Angle	Sattelites	Speed	HDOP
No	*0x80000000*	*0x80000000*	*0x8000*	*0xFFFF*	*0xFF*	*0xFFFF*	*0xFF*
Yes	*GPS_Longitude*	*GPS_Latitude*	*GPS_Altitude*	*GPS_angle*	*GPS_sattelites*	*GPS_Speed*	*GPS_HDOP*

1.4 *Record example*

Record example with explanations is shown below.



1.5 *Extended records*

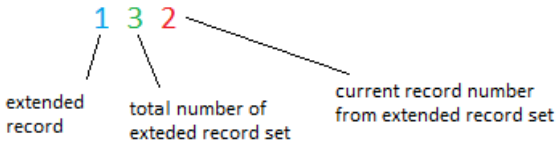
The size of all possible IO events is larger then a single record maximum size. That is why there is a technique to add many IO events in several records (with same header). Time stamp extension value must be interpreted as a decimal number of three digits – [a b c]. Each one of the 3 digits represents different information.

a) First digit (a) can be 0 or 1. Number 1 means that the record size of the current time stamp was exceeded and the records should be merged. Number 0 means that the record size of the current time stamp was not exceeded and no data have to be merged.

b) Second digit (b) can vary from 0 to 9. This digit represents how many records have to be merged of the current time stamp. For ex.: 0 means that there are total number of 1 record which has to be merged. Number 9 means that there are total number of 10 records which have to be merged. The maximum number of 10 records can be merged.

c) Third digit (c) can vary from 0 to 9. This digit represents which record of the current time stamp is of total (b) number of records. 0 means the first record. 9 means 10th record.

Example number of time stamp extension (0x84 = 132):



1.6 *Record summary*

Record structure summary is shown in the table below.

Group	Parameter	Size	Description
Time	Time stamp	4 Bytes	UNIX time stamp
	Time stamp extension	1 Byte	Virtual milliseconds, enables to identify multiple records in one second. If three records are collected in one second first record will have time extension 00, second record will have time extension 01, third record will have time extension 02.
	Priority	1 Byte	High/low
GPS/GLONASS	Longitude	4 Bytes	longitude value multiplied by 10000000
	Latitude	4 Bytes	latitude value multiplied by 10000000
	Altitude	2 Bytes	altitude value multiplied by 10
	Angle	2 Bytes	angle multiplied by 100
	Satellites	1 Byte	number of satellites in use
	Speed	2 Bytes	speed over ground km/h
	HDOP	1 Byte	HDOP multiplied by 10
	IO Data caused record	1 Byte	IO Data ID witch caused record
IO element	No. of IO data 1Byte	1 Byte	Number of IO data elements witch value is 1 Byte length
	IO Data ID	1 Byte	IO Data ID witch value is 1 Byte
	IO Data value	1 Byte	IO Data value witch value is 1 Byte length

	...	...	
	No. of IO data 2Byte	1 Byte	Number of IO data elements witch value is 2 Byte length
	IO Data ID	1 Byte	IO Data ID witch value is 2 Byte length
	IO Data value	2 Bytes	IO Data value witch value is 2 Byte length
	...	...	
	No. of IO data 4Byte	1 Byte	Number of IO data elements witch value is 4 Byte length
	IO Data ID	1 Byte	IO Data ID witch value is 4 Byte length
	IO Data value	4 Bytes	IO Data value witch value is 4 Byte length
	...	...	
	No. of IO data 8Byte	1 Byte	Number of IO data elements witch value is 8 Byte length
	IO Data ID	1 Byte	IO Data ID witch value is 8 Byte length
	IO Data value	8 Bytes	IO Data value witch value is 8 Byte length
	...	...	

2 Extended Protocol Records

Following chapter will cover only fields that are different from original record protocol described in Chapter 1.

2.1 *Record structure*

All records have defined structure which can vary in length. Record consists of 2 parts: header with fixed length (25 bytes) and body with varying length (4-101 bytes). Maximum record size is 126 bytes.

Header [25 B]	Body [4-101 B]

All data are in hex format.

2.2 *Record header*

All headers have the same parameters' fields. These fields are showned below.

Time stamp [4B]	Time stamp extension [1B]	Record extension [1B]	Priority [1B]	Longitude [4B]	Latitude [4B]	Altitude [2B]	Angle [2B]	Satellites [1B]	Speed [2B]	HDOP [1B]	Event ID which generates record [2B]

2.2.1 Time stamp extension

Time stamp extension – an extra byte to separate records with same time stamp. If some records have same time stamp when time stamp extension will increase starting with zero <0x00>. If there are no records with same time stamp parameter will always be zero. Parameter length – 1 byte.

See more: *2.4 Extended protocol extended records* (page 9).

2.2.2 Record extension

Record extension – an extra byte to separate records with same time stamp and time stamp extension. If some data does not fit into one record it is indicated by Record extension field. Parameter length – 1 byte.

See more: *2.4 Extended protocol extended records* (page 9).

2.2.3 Event ID which generates record

Parameter indicates why the record was created. Value is event ID number. Parameter length – 2 bytes (big endian).

2.3 *Record body*

Record body can vary in length. Length depends on configuration of the FM device. Body is devided into segmens of different length parameters (1, 2, 4 and 8 bytes long). Every segment holds the number of parameters with same size and ID numbers. For extended record protocol IO ID numbers are 2 bytes long (big endian).

2.4 *Extended protocol extended records*

The size of all possible IO events is larger then a single record maximum size. That is why there is a technique to add many IO events in several records (with same header). Record extension value must be interpreted as a hexdecimal number of two digits – [0xmn]. Each one of the 2 BCD digits represents different information.

a) First BCD digit (m) can vary from 0 to 7. This digit represents how many records have to be merged of the current time stamp. For ex.: 0 means that there are total number of 1 record which has to be merged. Number 7 means that there are total number of 8 records which have to be merged.

b) Second BCD (n) can vary from 0 to 7. This digit represents which record of the current time stamp is of total (m) number of records. 0 means the first record. 7 means 8th record.

Example number of time stamp extension:

- 0x74 => 5th record of 8 records set;
- 0x20 => 1th record of 3 records set;
- 0x00 => 1th record of 1 record set (this is not extended record);

2.5 *Extended Protocol Record summary*

Record structure summary is shown in the table below.

Group	Parameter	Size	Description
Time	Time stamp	4 Bytes	UNIX time stamp
	Time stamp extension	1 Byte	Virtual milliseconds, enables to identify multiple records in one second. If three records are collected in one second first record will have time extension 00, second record will have time extension 01, third record will have time extension 02.
	Record extension	1 Byte	Extended record indicator
	Priority	1 Byte	High/low
GPS/GLONASS	Longitude	4 Bytes	longitude value multiplied by 10000000
	Latitude	4 Bytes	latitude value multiplied by 10000000
	Altitude	2 Bytes	altitude value multiplied by 10
	Angle	2 Bytes	angle multiplied by 100

	Satellites	1 Byte	number of satellites in use
	Speed	2 Bytes	speed over ground km/h
	HDOP	1 Byte	HDOP multiplied by 10
	IO Data caused record	2 Byte	IO Data ID witch caused record
IO element	No. of IO data 1Byte	1 Byte	Number of IO data elements witch value is 1 Byte length
	IO Data ID	2 Byte	IO Data ID witch value is 1 Byte
	IO Data value	1 Byte	IO Data value witch value is 1 Byte length
	...	...	
	No. of IO data 2Byte	1 Byte	Number of IO data elements witch value is 2 Byte length
	IO Data ID	2 Byte	IO Data ID witch value is 2 Byte length
	IO Data value	2 Bytes	IO Data value witch value is 2 Byte length
	...	...	
	No. of IO data 4Byte	1 Byte	Number of IO data elements witch value is 4 Byte length
	IO Data ID	2 Byte	IO Data ID witch value is 4 Byte length
	IO Data value	4 Bytes	IO Data value witch value is 4 Byte length
	...	...	
	No. of IO data 8Byte	1 Byte	Number of IO data elements witch value is 8 Byte length
	IO Data ID	2 Byte	IO Data ID witch value is 8 Byte length
	IO Data value	8 Bytes	IO Data value witch value is 8 Byte length
	...	...	

3 FM & Server protocol

3.1 *Protocol Structure*

There is a standard Ruptela protocol for communication between FM device and server. All messages between server and device are sent by this standard. General protocol description is shown below.

Device to server:

Field	Packet length	IMEI	Command ID	Payload Data	CRC16
Size (bytes)	2	8	1	[1-1009]	2

Server to device:

Field	Packet length	Command ID	Payload data	CRC16
Size (bytes)	2	1	[1-1018]	2

Remark:

Difference is that server does not have IMEI field.

Maximum data packet size is 1 kB (1024 bytes).

3.1.1 Packet length

Parameter indicates the size of all packet size: all fields except itself and CRC16.

3.1.2 IMEI

IMEI is 64 bit variable. It is unique for every FM device.

3.1.3 Command

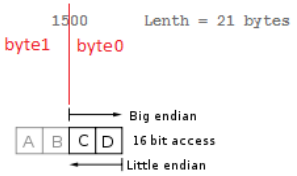
Command ID is 8 bit variable. Command ID describes what type of command is received. It how to handle DATA field.

3.1.4 Payload

All data which is sent.

3.1.5 CRC16

CRC is calculated using CRC-CCITT (Kermit) algorithm (http://www.lammertbies.nl/comm/info/crc-calculation.html). Parameter length – 2 bytes. Format is big endian. That is why bytes 0 and 1 are “switched”.



Endian	First byte (lowest address)	Middle bytes	Last byte (highest address)	Decimal 1000 (hexadecimal 3E8) in two bytes
big	*most* significant	...	*least* significant	03 E8
little	*least* significant	...	*most* significant	E8 03

CRC calculation algorithm is shown below (C programming language).

```
/*----------------------------------------------------------------------------
 *     FUNCTION: CRC16
 *----------------------------------------------------------------------------*/
unsigned short crc_16_rec (unsigned char *pucData, unsigned short ucLen) {
            //---------------------------------------------------------------
            unsigned int i;
            unsigned char ucBit, ucCarry;
            //---------------------------------------------------------------
            unsigned short usPoly = 0x8408;//reversed 0x1021
            unsigned short usCRC  = 0;
            //---------------------------------------------------------------
            for (i = 0; i < ucLen; i++) {
      usCRC ^= pucData[i];
      for (ucBit = 0; ucBit < 8; ucBit++) {
         ucCarry = usCRC & 1;
         usCRC >>= 1;
         if (ucCarry) {
            usCRC ^= usPoly;
         }
      }
   }
            //---------------------------------------------------------------
            return usCRC;
            //---------------------------------------------------------------
}
```

3.2 *Communication commands*

Communication between FM device and server always is initiated by FM device. There is a possibility to force FM device to connect to server by sending SMS message (see *SMS commands: connect, econnect*).

There is always a respone message from receiver.

3.2.1 Command 1/100 – records

FM device sends records according to the format which is shown below.

Field	Packet length	IMEI	Command ID	Payload			CRC16
				Records left	Number of records	Data (see *Records*)	
Size (bytes)	2	8	1	1	1	[1-1009]	2
Example hex	0113	00000B1A29F64B1A	01	00	0C	12 records data...	ADE9
Example dec	275	12207001062170	1	0	12	12 records data...	44521

Records left flag - Two possible values: 0 or 1. 0 – there are no records left in flash, 1 – there are records left in flash

Number of records - Describes how many records are in Record DATA field.

Data - Records with variable length. Record consists of non variable length field: time stamp, priority, GPS elements and variable length field: IO elements.

FM device uses command 1 (0x01).

Answer from server:

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	64	01	13BC
Example dec	2	100	1	5052

Server should use command 100 (0x64) for records response (acknowledge). Sample record packet is parsed in *PRO3 records packet parsing.txt* file.

Example of records data packet (raw data 825 bytes):

033500000C076B5C208F01011E5268CEF20000196E3A3A0AEF3E934F3E2D78000000000700000000005268CEFD0000196E3A3A0AEF3E934F3E2D780000000007000000005268CF080000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF130000196E3A3A0AEF3E934F3E2D780000000007000000005268CF1E0000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF290000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF340000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF3F0000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF4A0000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF550000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF600000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF6B0000196E3A3A0AEF3E934F3E2D78000000000700000000005268CF730000196E36630AEF42CE

4F6D0BF4040002220800000005268CF7E0000196E36B60AEF42BE4F6D0BF40000000007000000005268CF890000196E36B60AEF42BE4F6D0BF40000000007000000005268CF940000196E36B60AEF42BE4F6D0BF40000000007000000005268CF9F0000196E36B60AEF42BE4F6D0BF40000000007000000005268CFAA0000196E36B60AEF42BE4F6D0BF40000000007000000005268CFB50000196E36B60AEF42BE4F6D0BF40000000007000000005268CFC00000196E36B60AEF42BE4F6D0BF40000000007000000005268CFCB0000196E36B60AEF42BE4F6D0BF40000000007000000005268CFD600000196E36B60AEF42BE4F6D0BF40000000007000000005268CFD70000196E3C710AEF5EFF4F690BF4040001170800000005268CFE20000196E3B980AEF601A4F690BF40000000007000000005268CFED0000196E3B980AEF601A4F690BF40000000007000000005268CFF80000196E3B980AEF601A4F690BF40000000007000000005268D00030000196E3B980AEF601A4F690BF40000000007000000005268D00E0000196E3B980AEF601A4F690BF40000000007000000005268D0190000196E3B980AEF601A4F690BF40000000007000000005268D0240000196E3B980AEF601A4F690BF4000000000700000000046E2

Packet length – 0x0335 = 821

IMEI – 0x00000C076B5C208F = 13226005504143

Command ID – 0x01 = 1

Records left in device's flash memory – 0x01 = 1

Number of records in packet – 0x1E = 30

30 Records...

CRC16 – 0x46E2 = 18146

3.2.2 Command 68/100 – extended protocol records

FM device sends extended protocol records according to the format which is shown below.

Field	Packet length	IMEI	Command ID	Payload			CRC16
				Records left	Number of records	Data (see Extended Protocol *Records*)	
Size (bytes)	2	8	1	1	1	[1-1009]	2
Example hex	0113	00000B1A29F64B1A	44	00	0C	12 records data...	ADE9
Example dec	275	12207001062170	68	0	12	12 records data...	44521

Records left flag - Two possible values: 0 or 1. 0 – there are no records left in flash, 1 – there are records left in flash

Number of records - Describes how many records are in Record DATA field.

Data - Records with variable length. Record consists of non variable length field: time stamp, priority, GPS elements and variable length field: IO elements.

FM device uses command 68 (0x44).

Answer from server:

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	64	01	13BC
Example dec	2	100	1	5052

Server should use command 100 (0x64) for records response (acknowledge).

3.2.3 Command 2/102 – Device Configuration Data

This command is used to configure FM device: send configuration packets & control messages (see *4 Configuration*). Data in payload field is terminated with <0x0D><0x0A>. Server uses command 102 (0x02).

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	Not fixed	2
Example hex	000C	66	236366675F676574400D0A	10F2
Example dec	12	102	#cfg_get@	4338

FM device uses command 2 (0x02) to send response.

Field	Packet length	IMEI	Command ID	Payload	CRC16
Size (bytes)	2	8	1	Not fixed	1
Example hex	0018	00000B1A29F64B1A	66	406366675f73746172742331300D0A	7220
Example dec	24	12207001062170	102	@cfg_sts#10	29216

3.2.4 Command 3/103 – Device Version Info

Server uses command 103 (0x67). Command asks for the information about current firmware, hardware versions.

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	none	2
Example hex	0001	67	-	17B9
Example dec	1	103	-	6073

FM device uses command 3 (0x03).

Field	Packet length	IMEI	Command ID	Payload	CRC16
Size (bytes)	2	8	1	Not fixed	2
Example hex	0020	000315A07F44865A	03	343530322C30302E30322E32372C323731 332C32312C31	4920
Example dec	32	868204004279898	3	4502,00.02.27,2713,21,1	18720

3.2.5 Command 4/104 – Device Firmware Update

Command is used to update FM device's firmware: send configuration packets & control messages (see *5 Firmware*). Data in payload field is terminated with <0x0D><0x0A>. Server uses command 104 (0x68).

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	[1-1019]	2
Example hex	000C	68	7C46555F535452542A0D0A	B66B
Example dec	12	104	|FU_STRT*	46699

FM device uses command 4 (0x04).

Field	Packet length	IMEI	Command ID	Payload	CRC16
Size (bytes)	2	8	1	[7-10]	2
Example hex	0012	000315A07F44865A	04	042A46555F4F4B7C0D0A	FEF0
Example dec	18	868204004279898	4	*FU_OK|	65264

3.2.6 Command 5/107 – Smart Card Data

Command 5 (0x05) is used by FM device to send smart card data (fragments of DDD file) to server.

Field	Packet length	IMEI	Command ID	Payload	CRC16
Size (bytes)	2	8	1	[1-512]	2
Example hex	0012	000315A07F44865A	05	Raw data segment of DDD file	13AE
Example dec	18	868204004279898	5	Raw data segment of DDD file	5038

Answer from server command 107 (0x6B).

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	6B	01	9074
Example dec	2	107	1	36980

ACK possible values: 0 – negative acknowledgment, 1 – positive acknowledgment, 2 – card data rejected.

The decision that whole *.DDD* file is sent should be done by server. Server checks all received smart card data size. If the data size is equal to *smart card size parameter* sent by command 6 (smart card data size and time stamp) that means the end of DDD file. All smart card data packets' payloads are equal to 512 bytes except the last one. It is also an indication for the last packet (end of file).

3.2.7 Command 6/107 – Smart Card Data Size and Time stamp

FM uses this command 6 (0x06) to send information about smart card *.DDD* file and the time stamp when the file was created.

Field	Packet length	IMEI	Command ID	Payload		CRC16
				Size	Time stamp	
Size (bytes)	2	8	1	2 + 4 = 6		2
Example hex	000B	000315A07F44865A	06	5428	4E9CAF2C	F5B3
Example dec	11	868204004279898	6	21544	1318891308	F5B3

Answer from server command 107 (0x6B).

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	6B	01	9074
Example dec	2	107	1	36980

ACK possible values: 0 – negative acknowledgment, 1 – positive acknowledgment, 2 – card data rejected.

3.2.8 Command 7/108 – SMS via GPRS

Command is used to send SMS message to device via GPRS. Message text in payload is the same as a message via GSM network. Command 108 (0x6C) is used by server.

Field	Packet length	Command ID	Payload*	CRC16
Size (bytes)	2	1	4	2
Example hex	000B	6C	20536574696F20322C31	EB3E
Example dec	11	108	Setio 2,1	60222

*no password was needed but still a space symbol (0x20) before gsminfo text is written.

Command 7 (0x07) is used by device so send a response back to server.

Field	Packet length	IMEI	Command ID	Payload*	CRC16
Size (bytes)	2	8	1	4	2
Example hex	0011	000315A07F440B1D	07	534554494f20636f6e6669677572617469 6f6e2064617461206f6b	341C
Example dec	17	868204004248349	7	SETIO configuration data ok	13340

3.2.9 Command 9/109 – Diagnostic Trouble Codes

Command is used to send vehicle Diagnostic Trouble Codes (DTCs). FM device initiates sending session and server has to acknowledge packets. FM device starts to send DTC when it founds any changes in vehicle's diagnostic trouble codes (at least one new trouble code should appear or existing code should disappear). DTCs read frequency depends on configuration. Command 9 (0x09) is used by FM device. All currently read DTCs gains status – current

(0x01). All others have DTC status – history (0x02).

Field	Packet length	IMEI	Command ID	Payload		CRC16
				Number of DTC in packet	DTCs	
Size (bytes)	2	8	1	1	19	2
Example hex	000B	00000B1A29F64B1A	09	02	FF4E9CAF2C07D608F11A1480BA015030303130FF4E9CAF2C07D608F11A1480BA025030303131	8C91
Example dec	11	12207001062170	9	1	*	35985

*There are two DTC packets in a payload *DTCs* field.

1st packet: Time stamp – 1318891308; Longitude – 13,1467505; Latitude – 43,7551290; DTC status – current DTC; DTC – P0010.

2nd packet: Time stamp – 1318891308; Longitude – 13,1467505; Latitude – 43,7551290; DTC status – history DTC; DTC – P0011.

Payload fields are described below.

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Number of DTC in packet</td><td>Indicates how many separate Diagnostic Troubles Codes are in payload's sub field “DTCs”.</td></tr>
<tr><td>DTCs</td><td>This field holds Diagnostic Trouble Codes in packets shown below.

DTC packet:

<table>
<tr><th>Field</th><th>Length (bytes)</th><th>Description</th></tr>
<tr><td>Reserved</td><td>1</td><td>Reserved. Default value: 0xFF.</td></tr>
<tr><td>Time</td><td>4</td><td>UNIX time stamp when the DTC was read.</td></tr>
<tr><td>Longitude</td><td>4</td><td>Longitude when DTC was read.</td></tr>
<tr><td>Latitude</td><td>4</td><td>Latitude when DTC was read.</td></tr>
<tr><td>DTC status</td><td>1</td><td>Indicates the status of particular diagnostic trouble code:

1 – current DTC (vehicle still have it).

2 – history DTC (vehicle had this DTC at specified time (field 'Time') in the past).</td></tr>
<tr><td>DTC (text)</td><td>5</td><td>Diagnostic trouble code in ASCII format (text).</td></tr>
</table>
</td></tr>
</table>

Server answer with command 109 (0x6D).

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	6D	01	C4A4
Example dec	2	109	1	50340

ACK field can have three possible values. 0 – negative acknowledgment, 1 – positive acknoledgement, 2 – request DTC data (start force DTC data sending).

3.2.10 Command 10/110 – Tachograph Communication

Command is used to start reading (tachgraph initialisation) tachgraph data via FM device. Command packet has extra fields compared to other FM – server commands. These fields in data packet from FM device are described below.

Field	Description
Status	bit field tachograph task status. Status first byte indicating which object from tachograph is available. First byte is bit-field value. Bits indication: 1 – tachograph available, 2 – Card in card slot 1 is available, 3 – Card in card slot 2 is available.
Command SubID	1 – Status response (long status) in DATA field ***(Reserved for future use).***
	2 – ATR response (***in case of error***).
	3 – APDU packet in SubID payload field.
	4 – Finished authenticate.
	5 – Tachograph read period.
	6 – Unused.
	7 – Process stop response (end communication with tacho)
	8 – Upload data to tachograph.
	9 – Reading Tacho Data process (for error status).
	10 – Reading Card Data from Tacho process (for error status).
Packet Status	0 – NACK (not acknowledged).

	1 – ACK (acknowledgment OK).
	2 – ERROR.
	3 – Task not created.
	4 – Timeout (response from tachograph).
	5 – SubID incorrect (not founded).
	6 – Another tacho data file is in flash. Task not created.
	7 – Response from server timeout.
	8 – no tacho task.
	9 – interface extender not available.
	10 – wrong communication settings in FM device.
	11 – all repeats are exceeded, read failed.
SubID payload	SubID payload data or nothing (if NACK of ERROR is in packet's *Status* field).

Tachograph read routine is initiated by server using command 110 (0x6E), SubID 2 (0x02). After this FM device takes control of communication and server needs just to respond correctly. According to *SubID* command value *SubID payload* data can give different kind of information.

Server command 110 (0x6E):

Field	Packet length	Command ID	Command SubID	Reserved	SubID command payload	CRC16
Size (bytes)	2	1	1	2	Not fixed	2
Example hex	0020	6E	02	0000	3B9A96C01031FE5D0064057B0102318090007601513 0B20B518CFB07	A416
Example dec	32	110	2	0000		42006

FM device command 10 (0x0A):

Field	Packet length	IMEI	Command ID	Status	Command SubID	Packet Status	Reserved	SubID command payload	CRC16
Size (bytes)	2	8	1	4	1	1	2	Not fixed	2
Example hex	0018	000315A07F440B1D	0A	00000000	03	01	0000	00A4020C020002	65D7
Example dec	24	868204004248349	10	0	3	1	0		26071

After successful initialisation FM device will inform server about tachograph packet data with command 12 (0x0C). Server should answer ACK message with command 111 (0x6F). *DDD* file packets is then send by FM device to server using command 11 (0x0B). Server should send ACK message (0x6F) to every FM device message. Flowchart for getting tachgraph data is shown in file: *Server-FM Tacho_authentication_and_read_Flowchart_02.png*.

More detailed explanation is in *DOC_Tacho read detailed protocol.xls* document (*confidential*).

3.2.11 Command 11/111 – Tachograph Data Packet

Command is used to send *.DDD* file packets from FM device to server. FM device uses command 11 (0x0B).

Field	Packet length	IMEI	Command ID	Packet index	Payload	CRC16
Size (bytes)	2	8	1	2	[1-1009]	2
Example hex	03FC	000315A07F440B1D	0B	0000	1st fragment of *.DDD* file data	B314
Example dec	1020	868204004248349	11	0	1st fragment of *.DDD* file data	45844

Example raw data of this packet:

03 FC 00 03 15 A0 7F 44 0B 1D 0B 00 01 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 02 15 53 56 54 20 30 36 20 20 30 30 31 36 20 30 30 30 FF FF FF FF 51 39 D5 11 00 00 00 00 51 39 D5 89 00 00 00 15 01 41 42 43 31 32 33 20 20 20 20 20 20 20 51 39 D3 FE 00 00 08 20 00 02 CD 22 D3 02 D4 22 D5 02 D9 22 DB A0 00 03 02 15 53 56 54 20 30 36 20 20 30 30 31 36 20 30 30 30 30 51 39 D3 81 01 15 00 00 00 00 02 15 53 56 54 20 30 36 20 20 30 30 31 36 20 30 30 30 51 39 D3 FE 01 15 00 00 00 00 02 15 53 56 54 20 30 36 20 20 30 30 31 36 20 30 30 30 51 39 D5 88 01 15 00 00 00 00 00 00 5A 7C D4 64 54 F9 6F FD 17 18 50 48 20 8D 99 69 74 A0 2A E7 17 82 B6 4D F1 99 25 DA E6 04 02 F6 48 D2 F9 AE 45 6D 94 09 BE 73 3A 3D 69 5A 16 DE 44 D6 B9 FE DE 9A B7 FC D8 7B AC 38 2D 28 59 4E 73 94 E0 54 41 09 DB F5 7A DF 57 58 6E BF 4A D9 20 9B B3 3C EC 61 F7 5C 82 47 49 80 49 ED 63 20 A8 45 25 92 6A 1F EF AB 92 37 CC 81 EC 19 78 16 7F DE BF F9 CA D0 54 19 BE 5F 65 87 F3 7D 56 A7 76 02 51 47 AA 7F 00 00 08 00 04 01 4D 75 73 74 65 72 66 72 61 75 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 4B 72 69 73 74 61 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 63 D5 B7 00 51 47 17 28 00 00 00 00 51 47 17 B1 00 00 00 00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 01 4D 75 73 74 65 72 66 72 61 75 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 4B 72 69 73 74 61 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 63 D5 B7 00 51 47 18 63 00 00 00 00 51 47 19 04 00 00 02 15 01 41 42 43 31 32 33 20 20 20 20 20 20 20 51 47 17 B0 00 01 4D 75 73 74 65 72 66 72 61 75 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 4B 72 69 73 74 61 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 63 D5 B7 00 51 47 2A 6F 00 00 02 00 51 47 2B 61 00 00 02 15 01 41 42 43 31 32 33 20 20 20 20 20 20 20 51 47 19 03 00 01 4D 75 73 74 65 72 66 72 61 75 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 4B 72 69 73 74 61 20 34 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 63 D5 B7 00 51 47 3D 3C 00 00 02 00 51 47 3E FB 00 00 08 15 01 41 42 43 31 32 33 20 20 20 20 20 20 20 51 47 2B 60 00 00 14 20 00 03 2B 23 2D 03 30 1B 31 23 33 03 7D 1B 80 23 81 03 CD 1B CE 03 D1 23 D5 A0 00 AB 31 A3 33 AB 80 A3 81 AB CE A3 D1 03 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 30 51 47 19 03 01 15 00 00 00 02 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 30 51 47 2B 60 01 15 00 00 00 02 01 12 33 31 30 30 30 30 30 30 30 36 31 33 30 30 30 30 30 51 47 3E FA 01 15 00 00 00 08 00 00 05 B5 2F B4 AF 69 5C 30 C0 82 D5 78

Answer from server command 111 (0x6F).

Field	Packet length	Command ID	ACK	Packet index	CRC16
Size (bytes)	2	1	1	2	2
Example hex	0004	6F	01	0000	71C1
Example dec	4	111	1	0	29121

3.2.12 Command 12/111 – Information Packet About Tachograph Data

Command is used to send information about tachgraph data packet (see 2.2.9). Device uses command 12 (0x0C).

Field	Packet length	IMEI	Command ID	Payload						CRC16
				Data storage type	Data size	Data read time stamp	Data period start	Data period end	Data CRC16	Packet index	
Size (bytes)	2	8	1	1	4	4	4	4	2	2	2
Example hex	000B	00000B1A 29F64B1A	0C	01	00013 880	521C67CE	5130B20B	518CFB07	2A56	FFFF	39B0
Example dec	11	122070010 62170	12	1	80000	1377593294	136214580 3	13681937 99	10838	65535	14768

FM device command 12 payload explanation:

Sub field	Description
Data storage type	1 – in internal flash memory, 2 – in SD card memory.
Data size	Tachograph data size in flash or SD card (internal FM device memory).
Data read time stamp	Time stamp when tachograph was read (UNIX time stamp).
Data period start	Start date when tachograph data was read (UNIX time stamp).
Data period end	End date when tachograph data was read (UNIX time stamp).
Data CRC16	CRC16 of tachograph data (*.DDD*) which was read.
Packet index	Packet index. If data upload starts from beginning then index is 0xFFFF. Otherwise last index is '*last_index*-1'.

Server responds with command 111 (0x6F).

Field	Packet length	Command ID	ACK	Packet index	CRC16
Size (bytes)	2	1	1	2	2
Example hex	0004	6F	01	FFFF	8179
Example dec	4	111	1	65535	33145

Extra fields are explained below:

Field	Description
ACK	Values: 1 – positive acknowledgment, 2 – data rejected & FM device will delete all information from its memory. Other values are reserved.
Packet index	Packet index for ACK. First packet (info about tacho data) has index FFFF (response with this index to first packet).

Packet index

Tachograph data trasnfer command 111 (0x6F) has packet index field. This index is used by server to indicate which tacho data packet is expected/acknowledged. Server receives packet with index 0xFFFF if new data download has been started. If communication between server and FM device is terminated (ex.: GPRS link was broken) when data download is still in progress (not overed). Next time FM device will send packet "*Information packet about tachograph data*" (command 12) packet index value will be '*last index-1*'. If Server respond to FM device with just received index (*last index-1*) then next packet which will be sent to server from FM device will be with index increased by 1. If server

respond with different index value (compared to received index value) then FM device will send requested index packet.

Data flow example between server and FM device is shown below (command Ids are bolded):

Action*	Full packet
Info: FM to Server	00 1E 00 03 15 A0 7F 44 0B 1D **0C** 01 00 01 B3 BA 51 9F 2F D3 51 30 B2 0B 51 8C FB 07 00 00 FF FF 87 D4
Server to FM	00 04 **6F** 01 FF FF 81 79
FM to Server	03 FC 00 03 15 A0 7F 44 0B 1D **0B** 00 00 …
Server to FM	00 04 **6F** 01 00 00 71 C1
FM to Server	03 FC 00 03 15 A0 7F 44 0B 1D **0B** 00 01 20 20 20 20 20 **…**
Server to FM	00 04 **6F** 01 00 01 60 48
…	**…**
…	**…**
FM to Server	03 FC 00 03 15 A0 7F 44 0B 1D **0B** 00 6D 21 21 21 **…**
Server to FM	00 04 **6F** 01 00 6D C9 22
Last packet: FM to Server	02 37 00 03 15 A0 7F 44 0B 1D **0B** 00 6E 02 15 53 ...
Last packet: Server to FM	00 04 **6F** 01 00 6E FB B9

*this data transfer flow happens after successful tachograph initialisation (commands 10/110).

3.2.13 Command 14/114 – Transparent Channel data

Command is used by FM device to send transparent channel data to server (command 14). Packet is defined in the table below. Timestamp (UNIX timestamp) field indicates when the data was received by FM device from RS232 interface. Device does not start sending other tunnel channel data until ACK (acknowledgement) is received from server (command 114). Server can send data anytime after the link is established with the FM device. If server send a packet with non emtpy payload FM device will transmmit payload content to RS232 device. Tco4 device supports dual tunnel channel mode: two channels can work independently.

FM device sends transparent packet with command 14 (0x0E).

Field	Packet length	IMEI	Command ID	Port ID	Reserved	Payload		CRC16
						Timestamp	Data	
Size (bytes)	2	8	1	1	2	4	[1-1004]	2
Example hex	11	000315A07F44865A	0E	01	0000	53EA01DF	65	763C
Example dec	17	868204004279898	14	1	0	1407844831	101	30268

Extra fields are explained below:

Field		Description
Port ID		Values: 0 – port A, 1 – port B, 2 – port C (4th generation).
Reserved		Reserved bytes for future. Should leave zeros.
Payload	Timestamp	Timestamp of the payload data.
	Data	Raw data from port.

Server sends Acknowledgement to FM device with payload data (command 114 = 0x72).

Field	Packet length	Command ID	Port ID	Reserved	Payload	CRC16
Size (bytes)	2	1	1	2	[0-1016]	2
Example hex	5	72	01	0000	66	19F0
Example dec	5	114	1	0	102	6640

As we can see from the examples (tables above) FM device sends data 'A' which was received from RS232 device (port B) with (command 14). Then server acknowledges this transmission (command 114) and additionally sends data 'B'. This packet is received by FM device and data 'B' is transmitted to RS232 device (port B). 4th generation devices can operate both ports simultaneously.

3.2.14 Command 15/115– Identification packet

Command is used by device to send identification packet (information) to server. Identification packet payload is

defined in the table below. Device does not start sending other data until ACK (acknowledgement) is received from server (command 115). Firmware (cmd 104) and configuration (cmd 102) commands still work even if no ACK is received. All other commands from server are discarded if identification packet is not acknowledged.

Identification packet payload for 3rd generation devices.

Field	Device type	Firmware version	IMSI code	GSM operator code	Distance coefficient	Time coefficient	Angle coefficient
Size (bytes)	4	8	8	4	4	4	2
Example (hex)	50723033	30302E30322E3435	0000DFC29FF68CE4	0000601A	000003E8	0000003C	003C
Example (dec)	1349660723	3472326097237849141	246027000384740	24602	1000	60	60
Example (acii)	**Pr03**	**00.02.45**	-	-	-	-	-

Identification packet payload for 4th generation devices.

Field	Device type	Firmware version	IMSI code	GSM operator code	Distance coefficient	Time coefficient	Angle coefficient
Size (bytes)	4	11	8	4	4	4	2
Example (hex)	50723034	30302E30312E30312E3030	0000DFC29FF68CE4	0000601A	000003E8	0000003C	003C
Example (dec)	1349660724	3472326097221070897	246027000384740	24602	1000	60	60
Example (acii)	**Pr04**	**00.01.01.00**	-	-	-	-	-

As you can see from the example, some parameters (**bold text**) must be interpreted as ASCII characters. All other parameters are hex numbers.

Difference between 3rd and 4th gen payloads is the length of firmware version parameter.

FM device (3rd generation) sends identification packet with command 15 (0x0F).

Field	Packet length	IMEI	Command ID	Payload	CRC16
Size (bytes)	2	8	1	34	2
Example hex	002B	000315A07F44865A	0F	5072303374302E30322E34350000DFC29FF 68CE40000601A000003E80000003C003C	F1B1
Example dec	43	868204004279898	15		61873

Server sends Acknowledgement to FM device with command 115(0x73).

Field	Packet length	Command ID	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	73	01	CB25
Example dec	2	115	1	52005

3.2.15 Command 16/116– A-GPS file packet

Command is used to send Assisted GPS file to device. Same commands is used to send information, to request inforamtion and to send data to device. Usualy device requests Assisted GPS file from server and server must respond to requested data, but it is also possible to request from device information about Assisted GPS file stored in device. Assisted GPS files could be found at : http://alp.u-blox.com/ .

Device Assisted GPS file information

Field	Packet length	IMEI	Command ID	Payload			CRC16
				AGPS file type	AGPS file date	AGPS file size	
Size (bytes)	2	8	1	1	4	4	2
Example hex	0012	000310F5611DE6BC	10	01	53FB1CCD	00014D44	561D
Example dec	11	863071012513468	16	1	1408965837	85316	22045

Server sends Assisted GPS file fragment.

Field	Packet length	Command ID	Payload					CRC16
			AGPS file type	AGPS file date	AGPS file size	AGPS file offset	AGPS file data	
Size (bytes)	2	1	1	4	4	4	Length is not fixed	2
Example hex	020E	74	01	53FB1CCD	00014D44	00000400	EFFF7FE98A1B1701B7...	5CE4
Example dec	526	116	1	1408965837	85316	1024		23780

Server sends Assisted GPS file information request.

Field	Packet length	Command ID	Payload				CRC16
			AGPS file type	AGPS file date	AGPS file size	AGPS file offset	
Size (bytes)	2	1	1	4	4	4	2
Example hex	E	74	0	0	0	0	C3BE
Example dec	14	116	0	0	0	0	50110

To both server commands device answers with device assisted GPS file information command.

AGPS file type – 1-day ; 2- day; 3-day; 5-day; 7-day; 10-day; 14-day differential Almanac corrections.

AGPS file date – UTC time when file was created.

AGPS file size – file size in bytes.

AGPS file offset – Assisted GPS file offset from beginning.

AGPS file data – Assisted GPS file data.

3.2.16 Command 17/117 – Set IO value

Command 117 (0x75) is used by a server to set FM device IO to a specific value. Usually it is used to reset specific IO value to zero.

Field	Packet length	Command ID	Payload		CRC16
			IO ID	IO value	
Size (bytes)	2	1	4	4	2
Example hex	0009	75	000000AF	00000064	39CC
Example dec	9	117	175	100	14796

After setting specific IO to a new value FM device sends acknowledgement that ID is 17 (0x11).

Field	Packet length	IMEI	Command ID	Payload	CRC16
				ACK	
Size (bytes)	2	8	1	1	2
Example hex	000A	000310F56139B984	11	00	9799
Example dec	10	863071014336900	17	0	38809

Possible ACK field values and their meanings:

- 0 – IO value was changed;
- 1 – FM device failed to change IO value;
- 2 – the value change for specified IO is not supported.

3.2.17 Command 21/121 – Accident records

FM device sends accident records according to the format which is shown below.

Field	Packet length	IMEI	Command ID	Payload			CRC16
				Record size	Records left	Accident records (see *below*)	
Size (bytes)	2	8	1	1	1	[0-1009]	2
Example hex	03FF	00000B1A29F64B1A	15	12	01	...	ADE9
Example dec	1023	12207001062170	21	18	1	...	44521

Record size – actual size of a single accident record.

Records left - two possible values: 0 or 1. 0 – there are no accident records to be sent, 1 – there are more accident records to be sent.

Accident record – record consists of timestamp, latitude, longitude, rpm, speed, engine state, brake pedal position and acceleration peadal position:

Field	Timestamp	Latitude	Longitude	RPM	Speed	Engine state	Brake pedal position	Acceleration pedal position
Size (bytes)	4	4	4	2	1	1	1	1
Example hex	5416F6E1	20A0D3E4	0F08B6E6	07D0	32	01	00	01
Example dec	1410791137	547410916	252229350	2000	50	1	0	1

FM device uses command 21 (0x15).

Answers from server:

- default answer;

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	79	01	3655
Example dec	2	121	1	13909

- answer with a request to delete accident records from flash memory.

Field	Packet length	Command	ACK	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	79	02	04CE
Example dec	2	121	2	1230

Server should use command 121 (0x79) for accident records response (acknowledge). If FM device does **not receive** acknowledgment within timeout, it **resends** the packet. When the **default answer** is received, FM device sends a **next** packet. When the answer with the **request to delete** records is received, FM device **removes all accident records** from a flash memory and **stops** the packet sending procedure.

3.2.18 Command 30/130 – Garmin Device Request Status

Command is used by server to get the status about Garmin device and to know if Garmin device is correctly connected to FM device. Server sends a request with command 130 (0x82).

Field	Packet length	Command ID	CRC16
Size (bytes)	2	1	2
Example hex	0001	82	A71A
Example dec	1	130	42778

FM device should answer with command 30 (0x1E).

Field	Packet length	Command ID	Payload (Status)	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	1E	01	1E08
Example dec	2	30	1	7688

Garmin status can have three different states:

0x01 - Garmin is not responding (not connected to FM device).

0x02 - Garmin is responding (connected to FM device).

0x03 - Garmin is responding and supports unicode protocol (connect3ed to FM device and is compatible with Garmin protocol).

3.2.19 Command 31/131 – Garmin Device Data

Commands are used for communication between server and Garmin via FM device. If Garmin is corrrectly connected to FM device then FM will act as a clear channel between Garmin and server. Everything what is in command packet payload will be sent to Garmin. Garmin uses FMI standard for communication ((http://developer.garmin.com/lbs/fleet-management/fmi-supported-protocols/)). So there should be FMI message in the payload in command 131 (0x83) packet (table below).

Field	Packet length	Command ID	Payload (Garmin device data – FMI message)	CRC16
Size (bytes)	2	1	Not fixed	2
Example hex	001D	83	10A1164000060000003530303031310000000000000000000DC1003	753C
Example dec	29	131		30012

Answer from Garmin is also a FMI message in payload in command 31 (0x83) packet.

Field	Packet length	IMEI	Command ID	Payload (Garmin device data – FMI message)	CRC16
Size (bytes)	2	8	1	Not fixed	2
Example hex	003A	00000B1A2A3C833E	1F	100602A100571003578B002500000B1A2A3C833E1F10A1164100060200003530303031310000000000000000D91003	C112
Example dec	58	12207005664062	31		49426

FM device just forward FMI messages between server and Garmin device. Server should format FMI packet correctly otherwise where will be no answer from Garmin and FM device.

3.2.20 Command 32/132 – Weighting system data

Commands are used for communication between server and weighting system via FM device. If weighting system is corrrectly connected to FM device (via RS232 interface) then FM will send system's data strings to server using command 32 (0x20). Additionally FM device puts „record data“ information. Example is showned below.

Field	Packet length	IMEI	Command ID	Record data	Payload (Weighting system data)	CRC16
Size (bytes)	2	8	1	14	Not fixed (min: 9; max 100 bytes)	2
Example hex	0033	00000B1A2A3C833E	20	5268CEF200196E3A3A0AEF3E0101	303030303030303330303a303030303030303730303a474c4153530D	9F91
Example dec	51	12207005664062	32		0000000300:0000000700:GLASS<0xOD>	40849

Record data field.

Record data		
Parameter	**Size (bytes)**	**Value**
Timestamp	4	Unix time-stamp
Longitude	4	Longitude value multiplied by 10000000
Latitude	4	Latitude value multiplied by 10000000
GPS fix	1	1 – if device has GPS fix, 0 – if not
PORT	1	0 – PORT A, 1 – PORT B

Server will send acknowledgement command 132 (0x84) to FM device.

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	Not fixed	2
Example hex	0002	84	01	FA25
Example dec		132	1	64037

3.2.21 Command 105 – Set Connection Parameters

Command is used to set temporary connection settings (connection IP address, port and protocol type). Server uses command 105 (0x69).

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	Not fixed	2
Example hex	0015	69	3139322E3136382E302E312C393031352C544350	1763
Example dec	21	105	192.168.0.1,9015,TCP	5987

FM device does not send a response. Next time (for one time) FM device will connect to server using received connection parameters.

3.2.22 Command 106 – FM device Odometer Set

Command 106 (0x6A) is used by server to set specific value of FM device's virtual odometer. Usually it is used to reset odometer value to zero (0x00000000).

Field	Packet length	Command ID	Payload	CRC16
Size (bytes)	2	1	4	2
Example hex	0005	6A	12345678	8AEB
Example dec	5	106	305419896	35563

FM device does not send a response. Next generated record will have new odometer value.

3.2.23 Command 120 – Generate Accident Records

Command 120 (0x78) is used by server to request FM device to generate accident records and store them in non volatile memory.

Field	Packet length	Command ID	CRC16
Size (bytes)	2	1	2
Example hex	0001	78	FFCF
Example dec	1	120	65487

FM device does not send a response.

3.2.24 Command 34/134 – SD card logging functionality

Command is used by server to request log entries from device and to control logging functionality.

ATTENTION: internal structure of this command is very similar to Command 1/100 (Records), but this command have SubCommand in payload section. SubCommand is used to separate log requests from control commands and standard protocol log entries from extended.

3.2.24.1 SubCommand 0 (0x00) – Requesting data from log

SubCommand is used by server to get the log records stored on SD card. Server sends a request with command 134 (0x86) and subCommand 0 (0x00).

Field	Packet length	Command ID	SubCommand ID	Start timestamp	End timestamp	Reserved	CRC16
Size (bytes)	2	1	1	4	4	2	2
Example hex	000C	86	0x00	553F44F2	553F45A6	0x00	B0FA
Example dec	12	134	0	1430209778	1430209958		45306

Start timestamp – UTC time to read log records FROM (UNIX time stamp).

End timestamp – UTC time to read log records TO (UNIX time stamp).

FM device sends log records according to the format which is shown below.

Field	Packet length	IMEI	Command ID	Payload				CRC16
				SubCommand ID	Records left	Number of records	Data (see *Records*)	
Size (bytes)	2	8	1	1	1	1	[1-1009]	2
Example hex	0113	00000B1A29F64B1A	01	40	00	0C	12 records data...	ADE9
Example dec	275	12207001062170	1	64	0	12	12 records data...	44521

SubCommadn ID – two possible values: 64 (0x40) – SD card log records in **standard** protocol; 128 (0x80) – SD card log records in **extended** protocol.

Records left flag – two possible values: 0 or 1. 0 – there are no log records left in selected time interval, 1 – there are log records left in selected time interval.

Number of records – describes how many records are in Record DATA field.

Data – records with variable length. Record consists of non variable length field: time stamp, priority, GPS elements and variable length field: IO elements.

When session is closed during log transmission to server (timeout), upon next session, regular records are sent. SD card log must be requested from server again, with a respect to what records we already have in server.

In case if device don't have any records in requested time interval it will sends packet with empty payload (*Records left flag* and *Number of records* fields will have values 0).

NOTICE: event ID field of each record in Data payload block – for all log records is 252

Answer from server:

Field	Packet length	Command	SubCommadn ID	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	86	01	C995
Example dec	2	134	1	51605

Server should use command 134 (0x86) with subCommand 1 (0x01) for log records response (acknowledge).

After that FM device will send next log records pack from requested time interval if unsent log records available or send packet with empty payload (*Records left flag* and *Number of records* fields will have values 0) to show that there is no log records to send.

3.2.24.2 SubCommand 2 (0x02) – Stop sending procedure

SubCommand is used by server to stop log records sending and clear requested time interval. Server sends a request with command 134 (0x86) and subCommand 1 (0x01).

Field	Packet length	Command ID	SubCommand ID	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	86	02	FB0E
Example dec	2	134	2	64270

There is no answer to this command from device.

3.2.24.3 SubCommand 16 (0x10) – Erase SD card data

SubCommand is used by server to erase whole SD card (used to initialize SD card or to switch between Event records and Log records mode, only same type of data can be stored on SD card at same time and for ex. if user need to enable SD card logging functionality and at least one event record present on the SD card – to prevent missing Event records FM device generates temporary error state so user will able to download all event records to server and then execute this command to enable SD card logging). Server sends a request with command 134 (0x86) and subCommand 16 (0x10).

Field	Packet length	Command ID	SubCommand ID	CRC16
Size (bytes)	2	1	1	2
Example hex	0002	86	10	C89D
Example dec	1	134	16	51357

FM device should answer with command 34 (0x22) with subCommand 16 (0x10).

Field	Packet length	IMEI	Command ID	Payload		CRC16
				SubCommand ID	Status	
Size (bytes)	2	8	1	1	1	2
Example hex	000B	00000B1A29F64B1A	22	10	01	85DF
Example dec	11	12207001062170	34	16	1	34271

Status field can have two different states:

0x00 – SD card erase error.

0x01 – SD card erase completed.

3.2.24.3 SubCommand 32 (0x20) – Enable/Disable SD card logging functionality

SubCommand is used by server to remotely enable or disable SD card logging functionality (for example if accident happend and to prevent log data overwriting).

NOTICE: This command only enable/disable logging temporary (until device restarts or profile reload happeds).

Server sends a request with command 134 (0x86) and subCommand 32 (0x20).

Field	Packet length	Command ID	Payload			CRC16
			SubCommand ID	Parameter set	Logging interval	
Size (bytes)	2	1	1	1	4	2
Example hex	0007	86	20	01	553F44F2	D247
Example dec	7	134	32	1	1430209778	53831

FM device should answer with command 34 (0x22) with subCommand 32 (0x20).

Field	Packet length	IMEI	Command ID	Payload		CRC16
				SubCommand ID	Status	
Size (bytes)	2	8	1	1	1	2
Example hex	000B	00000B1A29F64B1A	22	20	01	0A0A
Example dec	11	12207001062170	34	32	1	2570

Status field can have three different states:

0x00 – SD card logging stopped.

0x01 – SD card logging started.

0x02 – SD card logging can't start (problems with SD card or it contains regular records.

4 SMS

There is a possibility to send SMS message to FM device. FM device answers with SMS message too. This is an easy way to quickly interact with FM device. SMS messages are used:

- to get specific information from FM device;
- to reconfigure some parameters of FM device;
- to influent FM device work.

SMS structure: **"(password) (command) (command text)"**

Password, command and command text are separated by space symbol.

Password – if there is no password, then you need to write just space symbol before the command.

Usually command text parameters are separated by 'comma (,) symbol.

4.1 *Commands*

4.1.1 Coords – current coordinates

SMS message is used to get current GPS status. Response has 8 parameters.

Parameter	Description
Time	Current GMT date & time.
lat.	Current latitude.
long.	Current longitude.
alt.	Current altitude (meters).
sat.	Currently visible satellites.
dir.	Current angle.
hdop	Current HDOP level.
state	Current GPS/GLONASS state: 1-off, 2-on no fix, 3-on got fix, 4-not responding, 5-sleep, 6-disabled.

Example: *pass coords*

Response example: *2013-04-24 07:01, lat. 46.1443183, long. 11.881766, alt. 217.5, sat. 8, dir. 198.10, hdop 100, state 3*

4.1.2 Version – FM device version

SMS message is used to get current FM device version. Response has 5 parameters.

Parameter	Description
1	Bootloader version.
2	Firmware version.
3	Hardware version.
4	GSM signal level.
5	Voltage status: 0 – lower then 8 Volts (bad), 1- higher then 8 Volts (OK).

Example: *pass version*

Response example: *5402,00.02.15,1089,5,1*

4.1.3 Gsminfo – GSM/GPRS information

SMS message is used to get GSM and GPRS information. Response has 15 parameters.

Parameter	Title	Description
ST	Start Time	Date & time (GMT) from the last FM device reset/power ON.
GSM network		
OP	Operator	GSM operator number
lvl	Level	GSM signal level.
LAC	Location Area Code	16 bit number thereby allowing 65536 location areas within one GSM PLMN.
CID	Cell ID	A GSM Cell ID (CID) is a generally unique number used to identify each Base Transceiver Station (BTS) or sector of a BTS
FM device modem parameters (M)		
I	Initialization	The number of times when FM device has tried to initialize modem since Start Time (ST).
R	Reset	The number of resets of modem since Start Time (ST).
SP	Status Pin	The number of times when modem was turned ON and turned OFF unsuccessfully.
GPRS service		
GPRS	General Packet Radio Service	Status of GPRS. There are two possible values: 0 – no GPRS / 1 – attached to GPRS.

O	Opened	The number of opened GPRS sessions.
C	Closed	The number of closed GPRS sessions.
E	Error	The number of GPRS errors.
Link with server (LK)		
O	Opened	The number of opened links.
C	Closed	The number of closed links.
E	Error	The number of link errors,
TMO	Timeout	The number of server response timeout.
Reset		
RS	Reset	FM device last reset source. Possible causes: 08, 03, 01 – reset was because of modem power loss; 04 – reset because of watchdog; 10 – reset because of Firmware update;
P	Protocol	GPRS protocol version: 0 – standard protocol; 1 – extended protocol.

Example: *pass gsminfo*

Response example: *ST:2013.04.20 23:26:33; OP 22210,lvl 15,LAC 20030, CID: 28289; M:I 126, R 125, SP: 0; GPRS 0:O 64, C 0, E 248; LK:O 575, E 1, TMO 126; RS: 04; P 0*

4.1.4 Imei

SMS message is used to get device IMEI number.

Example: *pass imei*

Response example: *IMEI: 863071016796615*

4.1.5 Reset

SMS message is used to reset FM device.

Example: *pass reset*

Response example: *Resetting device*

4.1.6 Connect – custom connection

SMS message is used to force FM device to connect (for one time) to server with custom IP, port and protocol settings. FM device creates dummy record just with header part. Triggered event ID = 0 (zero). SMS has three parameters. SMS format: *pass connect IP,Port,Protocol*

Parameter	Description
IP	32-bit number, commonly known as an Internet Protocol address (xxx.xxx.xxx.xxx).
Port	16-bit number, commonly known as the port number (xxxxx).
Protocol	The principal of communication. There are two available protocols: TCP and UDP.

Example: *pass connect 192.168.0.1,7011,TCP*

Response example: *connection data ok*

4.1.7 Econnect – emergency custom connection

SMS message is used to force FM device to connect (for one time) to server with custom APN, user, pass, IP, port and protocol settings. FM device creates dummy record just with header part. Triggered event ID = 0 (zero). SMS has five parameters. SMS format: *pass econnect apn,apnLogin,apnPassword,IP,Port,Protocol*

Parameter	Description
APN	An Access Point Name (APN) is the name of a gateway between a GPRS (or 3G, etc.) mobile network and another computer network, frequently the public Internet.
User	User name for APN settings.
Pass	Password for APN settings.
IP	32-bit number, commonly known as an Internet Protocol address (xxx.xxx.xxx.xxx).
Port	16-bit number, commonly known as the port number (xxxxx).
Protocol	The principal of communication. There are two available protocols: TCP and UDP.

Example: *pass econnect apn,apnlogin,apnpass,192.168.0.1,7011,TCP*

Response example: *Emergency connection data ok*

4.1.8 Getapn – get APN parameters

SMS message is used to get APN (*APN)*, username (*USER*), password (*PSW*), IPs (*IP1, IP2*), ports (*Port1, Port2*) and protocol (*TCP/UDP*) settings (described in 3.1.6) from FM device.

Example: *pass getapn*

Response example: *APN: banga User: PSW: IP1: 92.62.134.38 Port1: 9021 IP2: 195.14.173.3 Port2: 9000 TCP/UDP: 0*

*TCP/UDP: 0 – TCP, 1 – UDP.

4.1.9 Setconnection – change connection configuration

SMS message is used to permanently change FM device configuration settings: APN, APN username, APN password, protocol, IP1, PORT1, IP2, and PORT2 (parameter description: 3.1.6).

SMS format: *pass setconnection apn,apnlogin,apnpassword,Protocol,IP1,Port1,IP2,Port2*

Example: *pass setconnection apn,apnlogin,apnpass,TCP,111.111.111.111,1111,222.222.222.222,2222*

Response example: *set connection data ok*

If one of the parameters should be preserved, then the specific location for the parameter should be filled with **old**. For example, ip1 and port1 should be preserved (old value should remain):

Example: *pass setconnection apn,apnlogin,apnpass,TCP,*old*,*old*,222.222.222.222,2222*

Response example: *set connection data ok*

If the message is not ending with port2, then those parameters, which are not mentioned in the message should not be changed. For example ip2 and port2 were not in the message:

Example: *pass setconnection apn,apnlogin,apnpass,TCP,111.111.111.111,1111*

Response example: *set connection data ok*

If configuration failed to set when FM device sends response: *Set connection data incorrect*

4.1.10 Switchip – switch primary IP and port

SMS message is used to change current primary IP and port (IP1, Port1 or IP2, Port2).

SMS format: *pass switchip X*

X – which IP and port should be primary

IP1 primary set example: *pass switchip 1*

Response example: *Setting primary IP OK*

IP2 primary set example: *pass switchip 2*

Response example: *Setting primary IP OK*

If the operation was unsuccessful then the answer is: *Setting primary IP FAIL*

4.1.11 Plock – lock/unlock IP and port parameters

SMS message is used to lock/unlock possibility of changing IP ir port parameters. Lock/unlock passwords are 32-bite long and are provided by Ruptela support. Be careful with this SMS command.

SMS format: *pass plock plockPassword*

Example: *pass plock aNmuxyBxxr83jumWuBkx1rxkq8eZeaeC*

Response example: *plock OK*

If the operation was unsuccessful then the answer is: *plock ERROR*

4.1.12 Setio – set outputs

SMS message is used to set Dout1 and Dout2 output level. Values: 0 – low, 1 – high, 2 – do not change.

Remember: Douts have to be connected to electric circuit correctly.

SMS format: *pass setio X1,X2*

X1 – state of Dout1

X2 – state of Dout2

Example: *pass setio 0,1*

Response example: *SETIO configuration data ok*

If configuration SMS is incorrect, device will response: *SETIO configuration data incorrect*

4.1.13 Getio – read inputs/outputs states

SMS message is used to get status about Dout1, Dout2, Din1, Din2, Din3, Din4, Ain1 and Ain2. Values: 1 – high, 0 – low. Analog inputs – millivolts.

SMS format: *pass getio*

Answer SMS format: *DIN1=X,DIN2=X,DIN3=X,DIN4=X,DOUT1=X,DOUT2=X,AIN1=Y,AIN2=Y*

X – Digital value: 1 – high, 0 – low.

Y – analog value in millivolts.

Example: *pass getio*

Response example: *DIN1=0,DIN2=1,DIN3=1,DIN4=1,DOUT1=0,DOUT2=0,AIN1=4210,AIN2=8600*

4.1.14 Delrecords – delete all records

SMS message is used to delete all records from internal flash memory FM device memory.

Example: *pass delrecords*

Response example: *All records deleted*

4.1.15 Modrev – modem revision

SMS message is used to get modem revision information. Answer SMS format: *Modem revision: 24_symbol_info*

Example: *pass modrev*

Response example: *Modem revision: 1137B06SIM900M64_ST*

4.1.16 Caninfo – can configuration info

SMS message is used to get information about CAN settings of FM3 (about CAN1 settings for FM4) device (only works with Tco devices). These settings are used to see exact CAN interface setup in configuration file (it doesn't necessarily reflect actual mode of operation). Answer SMS format: *CAN enable: X Manufacturer Y Type Z Active A*

Parameter	Description
CAN enable	0 – CAN is disabled / 1 – CAN is enabled, FMS standard mode / 2 – CAN is enabled, LCV mode / 3 – CAN is enabled, OBD mode / 4 – CAN is enabled, Tachograph mode.

<table>
<tr><td>Manufacturer*</td><td>Manufacture group of Light Commercial Vehicles (number value).
<table><tr><th>Value</th><th>Name</th></tr><tr><td>1</td><td>VAG</td></tr><tr><td>2</td><td>Mercedes</td></tr><tr><td>3</td><td>Citroen</td></tr><tr><td>4</td><td>Ford</td></tr><tr><td>5</td><td>Fiat</td></tr><tr><td>6</td><td>Opel</td></tr><tr><td>7</td><td>Renault</td></tr><tr><td>8</td><td>Toyota</td></tr><tr><td>9</td><td>FMS Tractor</td></tr></table></td></tr>
<tr><td>Type*</td><td>Type of Light Commercial Vehicle (number value).</td></tr>
<tr><td>Active</td><td>0 – CAN mode is silent mode, 1 – CAN mode is active mode</td></tr>
</table>

*see *LCV_select.txt* file in newest FM *Configurator* folder for up-to-date information.

Example: *pass caninfo*

Response example: *CAN enable: 2 Manufacturer 2 Type 1 Active 0*

* CAN mode is LCV, vehicle manufacture group Mercedes, type is merccedes1, can is in silent mode.

4.1.17 Cansinfo – dual can configuration info

SMS message is used to get information about CAN1 and CAN2 settings of FM4 device (only works with Tco devices). These settings are used to see exact dual CAN interface setup in configuration file (it doesn't necessarily reflect actual mode of operation). Answer SMS format: *CAN1 enable: X Manufacturer Y Type Z Active A; CAN2 enable: X Manufacturer Y Type Z Active A (see caninfo message for fields description)*

Response example: *CAN1 enable: 2 Manufacturer 2 Type 1 Active 0; CAN2 enable: 2 Manufacturer 2 Type 1 Active 0*

4.1.18 Fastsleep

SMS message is used to shorten sleep time period to 30 seconds (default: 10 minutes) for one time (current time). Usually this command is used just for testing purpose.

Example: *pass fastsleep*

Response example: *Fast sleep after 30 s*

4.1.19 Getsd – SD card info

SMS message is used to get information about SD card inserted into FM Tco or FM Pro device. There are 4 available answers.

SD card is inserted and used for records: *Using SD Card for Records. Size: sector_count x sector_size B, H: SDrecordHead, T: SDrecordTail*

SD card is inserted and used for log: Using SD Card for Log and working OK. Size: sector_count x sector_size B, H: SDrecordHead, T: SDrecordTail

SD card is inserted and used for log, but some errors occured: Using SD Card for Log but ERROR. Size: sector_count x sector_size B, H: SDrecordHead, T: SDrecordTail

Parameter	Description
sector_count	the number of sectors in SD card.
sector_size	the size (in bytes) of one sector in SD card.
SDrecordHead	SD card address of the last of record's end.
SdrecordTail	SD card address of beginning of the first record.

SD card is not inserted: *Using Internal Flash*

Example: *pass getsd*

Response example: *Using SD Card. Size: 3911680 x 512 B, H: 6008, T: 5993*

4.1.20 Clear obd – clear OBD values

SMS message is used with FM Tco OBD (00.03.XX) device. Purpose is to clear all OBD (On-board diagnostic) related data in device memory (not configuration). It can be assumed as OBD values reset command.

Example: *pass clear obd*

Response example: *OBD parameters and DTC cleared*

4.1.21 IEversion – TCO extender version

SMS message is used with FM Tco TCO (00.04.XX) device. Purpose is to get extender's (optional external FM gadget) version. Answer SMS format: *conf:X,ver:Y*

X – status of extender gadget: 0 – no extender / 1 – tachograph is connected to Port A via extender / 2 – tachograph is connected to Port B via extender / 3 – tachograph is connected to FM Tco TCO device CAN.

Y – extender firmware version (text).

Example: *pass ieversion*

Response example: *conf:1,ver:IE.00.01*

4.1.22 Tacho – tachograph status

SMS message is used get tachograph status information. Answer SMS format: *TACHO status:X*

Status (*X*)	Description
0	Not available.
1	Everything is OK.
2	Tacho parameters not configured.
3	Extender not responding.
4	Tacho task is in progress.
5	Physical communication OK, logical is not OK.

Example: *pass tacho*

Response example: *TACHO status:1*

4.1.23 webcoords – Google maps hyperlink with coordinates

SMS message is used to get hyperlink to Goolge maps with coordinates, current vehicle speed and ignition status.

Parameter	Description
Time	GMT date & time of coordinates
Hyperlink	Hyperlink to Google maps with coordinates: latitude and longitude
Speed	Current vehicle speed, km/h
Ignition	Ignition status

Syntax: *pass webcoords*

Can be 3 differrent SMS message responses.

1. When GPS data is available.

Response example: *2015-05-11 13:01, https://www.google.com/maps/?q=54.7404933,25.2222366, speed: 94, ignition: ON*

2. When GPS not available.

Response example: *GPS data not available. Ignition: ON*

3. When GPS not available but was available then device can return last known coordinates and the time shows when the coordinates were taken.

Response example: *No GPS. Last entry: 2015-05-16 17:32, https://www.google.com/maps/?q=54.7404933,25.2222366, speed: 65, ignition: ON, current ignition: OFF*

4.1.24 setiotime – set output for temporary period

Using this feature, FM is able to switch DOUT for temporary period of time. User must provide pulse lengths for logical '1' & '0' values.

SMS format example: **pass setiotime 1 500 0 500,0 200 1 300**

Description:

pass	SMS password
setiotime	Command Identifier
1	DOUT1 first logical state (1/0)
500	DOUT1 first logical state length (ms)
0	DOUT1 second logical state (1/0)
500	DOUT1 second logical state length (ms)
,	DOUT separator
0	DOUT2 first logical state (1/0)
200	DOUT2 first logical state length (ms)
1	DOUT2 second logical state (1/0)
300	DOUT2 second logical state length (ms)

Impulse resolution is 10 ms. Minimum impulse duration is 10 ms. If Eco-panel is connected, minimum resolution and dur ation is 50 ms.When the FM device receive the setiotime it stores the status of DOUT's and after the sequence restores the previous status. If one setiotime interrupts another, the state is **UNDEFINED** ('1' or '0') after both sequences are finished. '0' means **GND** and '1' means **no GND**.

If one of the DOUT is configured as LED, Buzzer blocking or Jamming block, setiotime is not possible to use.

Additional feature.

It is possible to set a repeat amount of a sequence.

SMS format: **pass setiotime 1 500 0 500 n=10,0 200 1 300 n=20**

Description:

pass	SMS password
setiotime	Command Identifier
1	DOUT1 first logical state (1/0)
500	DOUT1 first logical state length (ms)
0	DOUT1 second logical state (1/0)
500	DOUT1 second logical state length (ms)
n	Repetition identifier
=	Setter symbol
10	Amount of repetitions
,	DOUT separator
0	DOUT2 first logical state (1/0)
200	DOUT2 first logical state length (ms)
1	DOUT2 second logical state (1/0)
300	DOUT2 second logical state length (ms)

n	Repetition identifier
=	Setter symbol
20	Amount of repetitions

Each single DOUT has 10 slots for impulses levels (high / low). No more than 10 can be defined for one DOUT.

Longest time possible is 999 999 999 ms. Max number of repeats is 9999.

Max impulse count – as many as you can fit into 160 symbols of SMS.

It is possible to interrupt a sequence with “pass setiotime 0 10,0 10” followed by setio SMS command “pass setio 1,1”.

The sequence would be interrupted and the states of DOUT’s is defined that’s it.

It is possible to set only one of the DOUT.

SMS format example for set DOUT1: **pass setiotime 1 500 0 500**

SMS format example for set DOUT2: **pass setiotime ,0 200 1 300**

Response example: *setiotime set OK*

If configuration SMS is incorrect, device will response: s*etiotime syntax error in DOUTX settings: error text*

Where „DOUTX“ is DOUT1 or DOUT2. „Error text“ is described below in the table. It's minimal diagnostic when something wrong with impuse set in SMS.

Error text	Error description
other process controls output	Means that other functionality is configured on DOUT's. For example: LED or buzzer
no comma symbol	No comma separator. Comma is used to separate DOUT's configuration
wrong format	When was received not enough symbols
wrong level	Level can be only logical '1' or '0'. Other values is unacceptable
no space symbol	After level symbols ust to be space symbol
wrong ms number	Wrong time number
wrong repeat number	Wrong repeat count number
exceeded max slot	For one output is possible to set maximum 10 slot. If this count is exceeded this error is displayed
pulse is shorter than 10ms	Pulse length less than 10 miliseconds was set. It's not acceptable
other reason	Includes other reason not described in this table

4.1.25 Banned – temporary banned operators

SMS message is used with 4gen FM device. Purpose is to get information about temporary banned operators. Answer SMS format: *Already banned:X, Newly banned:Y, ops:ZZZ.*

Parameter	Description
X	Number of times when FM tried to ban operator which has already been in the banned list.
Y	Number of times when FM added operator in the banned list.
ZZZ	List of curently banned operators (which are still in the list).

Example: *pass banned*

Response example: *Already:1, Newly banned:1, ops:24602,*

4.1.26 accinfo

SMS message is used to check if accelerometer is calibrated for eco-driving functionality. Only state information is should be interpreted.

Parameter	Description
State	0 – not calibrated 1 – calibration started 2 – zero position calibration is in progress 3 – forward movement calibration paused 4 – forward calibration 5 – forward calibration in progress 6 – forward calibration in progress 11 – calibrated
	Number of times when FM added operator in the banned list.
*XYZo, N, ABC, N_, A_B_C_*	Parameters used for accelerometer events' calculations.

Example: *pass accinfo*

Response example: *AXL state:11; XYZo:0.0,0.0,1.0; N:-0.023; ABC:-0.15,0.9,0.32; N_:0.8; A_B_C_:1.24,-0.42,0.6*

4.1.27 accreset

SMS message is used to reset accelerometer (used for eco-driving functionality) calibration.

Example: *pass accreset*

Response example: *Acc reset OK*

4.1.28 lastchange

SMS message is used to know when was the last time device configuration or primary server IP changed.

Example: *pass lastchange*

If time is not synchronized, or changes occur during that time, SMS content will be:

Response example: *lastchange ip: no time available; cfg: no time available*

If time is available:

Response example: *lastchange ip: 2015-11-09, 09:02; cfg: 2015-11-09, 10:02*

Configuration change date and time will be updated during any type of configuration modification.

When device is connecting to GPRS and tries to open link (with different IP settings), IP change date and time will be updated. When device reconnects to server with same IP but different Port, IP change date and time will not be updated.

4.1.29 SMS during critical process

There are 4 critical processes:

- *Firmware update*
- *Configuration update*
- *Tacho read*
- *Smart Card read*

During these processes following SMS commands will be ignored:

- *reset*
- *connect*
- *econnect*
- *switchip*
- *setconnection*
- *delrecords*
- *setcfg*
- *getcfg*
- *setioparam*
- *getioparam*
- *clear obd*
- *tacho*

Response *example: The device is busy with critical process. Please try again later.*

4.1.30 *setcfg*

This SMS message is dedicated to change FM parameters. After SMS password, user inputs parameter ID and parameter value:

password **setcfg** ParamID1 Value1, ParamID2 Value2, ParamID3 Value3, …

Examples:

password setcfg 101 wave, 102 pioneer, 103 , 100 1
Sets APN name: wave, APN user: pioneer, no APN passsword, Protocol: UDP

password setcfg 4202 1, 4242 12, 4282 5, 4322 10, 4362 6
Sets IO in 4 profile, slot 2. IO modem temperature: enabled, level is 12, delta is 5, average is 10

Only parameters provided with SMS are changed, others remain the same.

One SMS message can be 160 symbols long, so the maximum simultaneously transmitted parameters count in SMS message are limited to one message length.

Setcfg command, allows user to change all parameters of the configuration except timetable (because it will not fit into one SMS message), passwords (configuration and SMS) and "enable SMS configuration".

Following responses are provided for the number, which sent configuration by SMS:

- When succesfully setting parameters for FM device: „Configuration parameter(s) was set!"
- Incorrect parameter setting: „*Configuration parameter(s) was NOT set! Parameter No. 1 is incorrect*".
- Unsuccesfull set-up, when parameters are locked: „*Configuration parameter(s) was NOT set! Parameter No. 1 is locked*"
- Unsuccesfull set-up, when setting IO parameters, but not every setting was provided by SMS: „*Configuration parameter(s) was NOT set! IO slot No. 9 is not fully set*".
- Unsuccesfull set-up, when device is busy: „*Configuration parameter(s) was NOT set! Device is busy try again later*"
- If SMS configuratio is disabled in configuration tool: „*You do not have permission to change the settings*".

SMS configuration feature is enabled/disabled in configuration tool, in Authorized numbers section, by putting a checkbox by the „Enable SMS configuration".

4.1.31 *getcfg*

Structure of getcfg SMS:

password **getcfg** id

This command is only used for getting the current status of the parameter selected.

Maximum simultaneously received parameters in one SMS message is limited by SMS message length (160 symbols).

· When parameter ID is correct, FM-device answer: „*ID: XXX,value:XXX;*“

Example: ‘*ID: 96,value:1000;*’

Value can be string or number.

Multiple parameter values can be requested in one SMS message and when response does not fit in one message then in the end of the message there will be a phrase: “*other values not fit*”.

Example:

Request:

“password getcfg 100,101,102,110,120,111,121,130”

Response:

“ID:100,value:1; ID:101,value:aerospace; ID:102,value:laguna; ID:110,value:101.16.17.245; ID:120,value:23451; ID:111,value:m2m.member.com; other values not fit”

· Parameters configuration password and SMS password are not reachable with this SMS. In this case, the device answer:

„*ERROR: parameter(s) read is forbidden*“

· If parameter ID is invalid, answer would be: „*ID:XXX,value:requested ID not found;*“

Example: “*ID:556,value:requested ID not found;*“

Other possible replies when something wrong:

“ERROR: request is empty”
“ERROR: wrong request syntax”
“ERROR: allowed numbers and commas only”
“ERROR: requested parameter ID too big”
“ERROR: requested parameter ID is low”

· If SMS configuration is disabled in configuration tool or configuration has password:

„*You do not have permission to read the settings*“

4.1.32 *setioparam*

Structure of setioparam SMS:

password **setioparam** **id**=*id,***profile**=*profile,***enable**=*enable,***level**=*level,***delta**=*delta,***average**=*average,***eventon**=*eventon,***include**=*include,***priority**=*priority,***switch**=*switch,***edge**=*edge*

Example:

01234567890123456 **setioparam**

id=256,**profile**=4,**enable**=1,**level**=3500,**delta**=250,**average**=2000,**eventon**=2,**include**=1,**priority**=1,**switch**=1,**edge**=3

This SMS can be used in any way, but it must at least contain ***Id***,***profile*** and ***enable*** parameters. Shortest possible message should be written this way:

password **setioparam id**=*id*,**profile**=*profile*,**enable**=*enable*

Example:

01234567890123456 **setioparam id**=256,**profile**=4,**enable**=1

If the short version is used the other parameters are set with old values. ***Id***,***profile*** and ***enable*** parameters are mandatory to make a valid SMS command.

Note that only two spaces between *password*[space]**setioparam**[space]***id*** are required. All other commands are separated by comma.

SMS can be written in lowercase and uppercase letters. Therefore, **setioparam** SMS is not case sensitive.

If SMS command doesn't define new values to **level**, **delta**, **average**, **eventOn**, **include**, **priority**, **switch** or **edge** parameters, it will not be changed. Previous value will remain.

When the FM-device receives the SMS where the parameter **enable** is set to 1, it searches in the current configuration if this parameter with the same ID is already enabled.

- If parameter with the same ID was not enabled, FM-device will set this IO parameter to the first free slot that is available.

- If IO parameter with the same ID was enabled, FM-device will overwrite the IO parameter with new values.

- If FM-device finds more than one IO parameters with the same ID it will send an error message.

If newly created IO event is not set to specific parameters it will be set to default. Default values are listed below:

Level	Delta	Average	EventOn	Include	Priority	Switch	Edge
0	0	1000	2	0	0	0	0

For set **enable**, **eventOn**, **include**, **priority, switch oe edge** it is necessary to indicate the state with a number:

Enable	EventOn	Include	Priority	Switch	Edge
0 – disable 1 – enable	0 – Hysteresis 1 – Change 2 – Monitoring	0 – not include data 1 – include data	0 – Low 1 – High	0 – no switch 1 – 1st profile 2 – 2nd profile 3 – 3rd profile 4 – 4th profile	0 – On Both 1 – On rising 2 – On falling

Following responses are provided for IO configuration by SMS:

· If IO parameter was set correctly, the FM-device would answer:

„*setioparam OK, slot: XX*“

Where ‘slot’ is the slot number where the FM-device set parameter.

· If IO parameter was not set correctly, FM-device would answer:

„*setioparam ERROR, <explanation>*”

· If all slots are full, FM-device would answer:

„*setioparam ERROR, no free slots for set the I/O*“

· If FM-device finds more than one IO parameters with the same ID it would answer:

“*setioparam ERROR, more than one I/O with same ID*”

· If user sent a SMS to disable (enable field = 0), and the parameter is not found as enabled, FM-device would answer:

„*setioparam ERROR, parameter is already disable*“

· If SMS configuration is disabled in configuration tool:

„*You do not have permission to change the settings*“

4.1.33 *getioparam*

Structure of getioparam SMS:

password **getioparam** id,profile

This command is only used for get the current status of the IO parameter selected.

· If IO parameter was enabled, the FM-device would answer:

Example:
“*id=28,profile=3,enable=1,level=0,delta=0,average=1000,eventon=2,include=0,priority=0,switch=0,edge=0*”

· If IO parameter was not enabled, the FM-device would answer:

„*I/O ID XXX is NOT enabled*“

· If IO parameter ID is invalid, the FM-device would answer:

„*ERROR I/O ID XXX does not exist*“

· Other possible replies when is something wrong:

“ERROR: I/O ID XXX read is forbidden”
“ERROR: more than one I/O parameter with same ID was found”
“ERROR: wrong request syntax”

Where XXX – IO ID

· If SMS configuration is disabled in configuration tool or configuration has password:

„*You do not have permission to read the settings*“

4.1.34 *setvalue – set specific IO values*

Purpose is to set specific IO values.

For now, this command is supported by IO parameters:

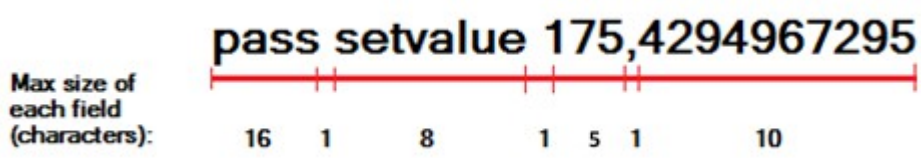
- Virtual odometer (ID:65)

- ECO Driving absolute idling time (ID:175).

- CAN Bus Distance (ID:114). Value will be overwritten by valid message from CAN-Bus (if available in specific vehicle).

In general SMS structure is: “*(password) (command) (command text)*”

In “command text” field there will be ID and its value separated by comma (,).



The response format depends on these options:

1. If IO value was changed successfully, then format is: ID,value
2. If FM device failed to change IO value, then format is: ID,fail
3. If the value change for specified IO is not supported, then format is: ID,unsupp
4. If the FM device is even failed to parse the ID or data is incorrect: Set IO value data incorrect

Example: *pass setvalue 65,0* Response example: *65,0*

Example: *pass setvalue 175,0* Response example: *175,0*

Example: *pass setvalue 155,15* Response example: *155,unsupp*

4.1.35 Supported SMS commands table

	Eco3	Pro3	Tco3-OBD	Tco3-TCO	Eco4	Pro4	Lcv4	Tco4	Plug
accinfo	●	●	●	●	●	●	●	●	●
accreset	●	●	●	●	●	●	●	●	●
banned		●	●	●	●	●	●	●	●
caninfo		●	●	●		●	●	●	
cansinfo						●	●	●	
clear obd			●				●	●	●
connect	●	●	●	●	●	●	●	●	●
coords	●	●	●	●	●	●	●	●	●
delrecords	●	●	●	●	●	●	●	●	●
econnect	●	●	●	●	●	●	●	●	●
fastsleep	●	●	●	●	●	●	●	●	●
getapn	●	●	●	●	●	●	●	●	●
getcfg					●	●	●	●	
getecu			●				●	●	
getio	●	●	●	●	●	●	●	●	
getioparam					●				
getioparam					●	●	●	●	
getsd			●	●		●	●	●	
gsminfo	●	●	●	●	●	●	●	●	●
ieversion				●					
imei	●	●	●	●	●	●	●	●	●
lastchange								●	
modrev	●	●	●	●	●	●	●	●	●
optiver				●				●	
plock	●	●	●	●	●	●	●	●	●
reset	●	●	●	●	●	●	●	●	●
setcfg					●	●	●	●	
setconnection	●	●	●	●	●	●	●	●	●
setio	●	●	●	●	●	●	●	●	
setioparam					●	●	●	●	
setiotime					●				
setvalue						●	●	●	
switchip	●	●	●	●	●	●	●	●	●
tacho				●				●	
version	●	●	●	●	●	●	●	●	●
webcoords					●	●	●	●	●

4.2 *Informational messages, alerts*

4.2.1 Driving rule violation, accident

SMS message is sent if one of the following SMS alerts are configured: **overspeeding, harsh braking, extreme braking, harsh acceleration, DIN1, DIN2, DIN3, DIN4**. Purpose of this message is to inform about the driving rule violations or accidents.

Message format:

[violation type/accident type] (**count**)

…

...
[violation type/accident type] (**count**)

count – number of accidents or driving rule violations between messages.
Message may contain **single** or **multiple** alerts.

Violation/accident types:

- Over speeding;
- Extreme braking;
- Harsh braking;
- Harsh acceleration;
- DIN1;
- DIN2;
- DIN3;
- DIN4.

Examples:

Overspeeding (5)
DIN1 (2)
Harsh braking (8)

4.2.2 SMS alerts with date & time

SMS message can be sent if one of the following SMS alerts are configured: **overspeeding, power supply disconnecting, DIN1, DIN2, DIN3, DIN4 (ignition)**. Purpose of this message is to inform when these events were triggered. Date & time is GMT. To receive SMS alerts, "SMS Alert Number must be configured.

Message format:

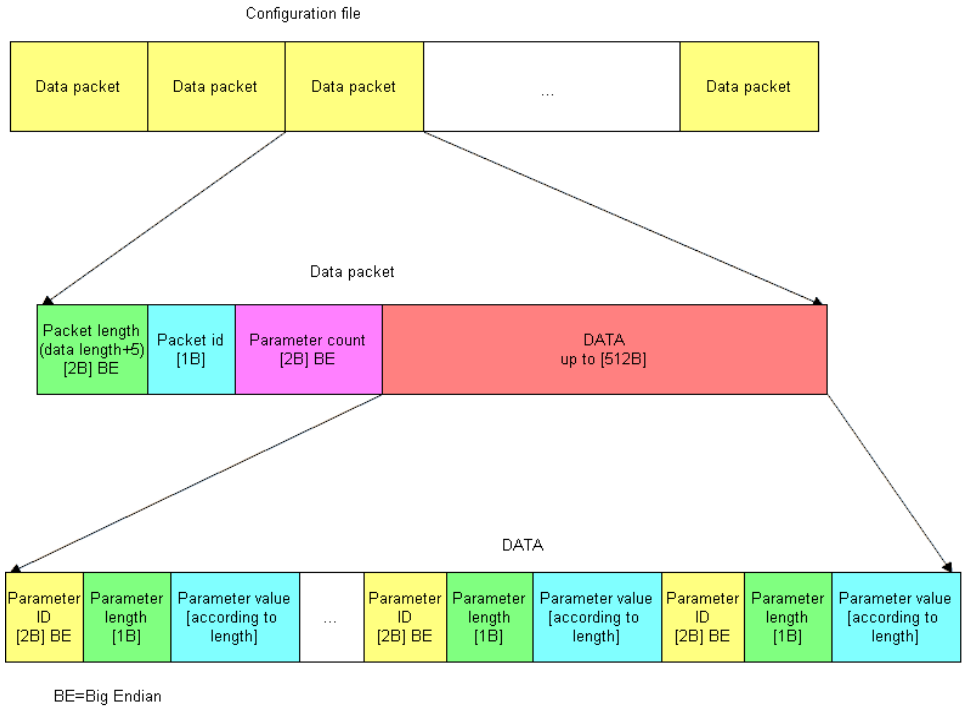
date1&time1-event1; date2&time2-event2; …

Examples:

2015.05.01 16:24:01-device disconnected;
2015.06.02 22:05:16-ignition: ON; 2015.06.02 22:05:35-DIN1: OFF;
2015.06.14 08:30:45-overspeed;

5 Configuration

There are two possible ways to read/write configuration from/to FM device. One option is to use USB connection with PC (personal computer). Another option is to use GPRS service (air). There are specific rules how to download/upload configuration. Configuration file has extensions *.ft3c* (Tco), *.fp3c* (Pro) and *.fe3c* (Eco). Configuration file is made of data packets:



5.1 *Configuration data packet*

Configuration packet has defined structure.

Configuration packet length [2B] big endian	Packet ID [1B]	Parameters count [2B] big endian	Configuration data [4-512 B]			
			Parameter ID [2B] big endian	Parameter length [1B]	Parameter value [1-128 B]	...

*big endian means that the value should be interpretated

Parameters description.

Parameter	Description
Configuration packet length	The length (in bytes) of current configuration data packet (with unique packet ID). 2 bytes long.
Packet ID	Unique packet identifier of current configuration file. 1 byte long.
Parameter count	The number of parameters in configuration data sector. 2 bytes long.
Configuration data	

Parameter ID	Unique parameter identifier. 2 bytes long.
Parameter length	The length of parameter's (defined in *Parameter ID* field) value. 1 byte long.
Parameter value	Actual parameter value. Length is defined in *Parameter length* field.

Usually there are several dozens of parameters (configuration data) in every configuration packet. There is a simple tool to check configuration file. *ConfigParser* can parse whole raw configuration file hex string (instructions are in *explain.txt* file).

Example of configuration data packet1 (raw data 510 bytes):

fe01011f0064000101650020004f1c000008080007000200070108000005f1c00feffffffffffff7fffffff7f66002000da180011d50a3a03000200c9010a00a0b
b18007300740065006d0000006100670020000da180011d50a3a03000200c9010a00a0bb18007300740065006d00000061006e0028006a1a000008080
013000200d1010d0000000000e04fd020ea3a6910a2d808002b30309d19002f436f002800591c001934e089160002005a010800a0d01c00a000000078
0f1800f8c0387b0010377bdd070006780002000079000200007a00023c008200100037303635393233353131006e6700008300100037303635393030
333933006e6700008400100000000000000000000000000000000000000850010000000000000000000000000000000008600100000000000000000000
0000000000087001000000000000000000000000000000000000880010000000000000000000000000000000000890010000000000000000000000000
0000008a00100000000000000000000000000000000008b00100000000000000000000000000000000008c001000831c0011d50a3a11000200b9000800
8d0010004f1c000008080007000200070108000c8000100e80304000000000e90304000000000ea0304000000000eb0304000000000ec0304000000000ed030
400000000ee0304000000000ef0304000000000f0030400000000

Packet1 (parsed):

fe01 - packet length (510 bytes)
01 - Packet ID (1st)
1f00 - Parameter count (31)
----- 31 parameters -----
ID length value

6400 01 01
6500 20 004f1c000008080007000200070108000005f1c00feffffffffffff7fffffff7f
6600 20 00da180011d50a3a03000200c9010a00a0bb18007300740065006d0000006100
6700 20 00da180011d50a3a03000200c9010a00a0bb18007300740065006d0000006100
6e00 28 006a1a000008080013000200d1010d0000000000e04fd020ea3a6910a2d808002b30309d19002f43
6f00 28 00591c001934e089160002005a010800a0d01c00a0000000780f1800f8c0387b0010377bdd070006
7800 02 0000
7900 02 0000
7a00 02 3c00
8200 10 00373036353932333531310 06e670000
8300 10 00373036353930303339330 06e670000
8400 10 00000000000000000000000000000000
8500 10 00000000000000000000000000000000
8600 10 00000000000000000000000000000000
8700 10 00000000000000000000000000000000
8800 10 00000000000000000000000000000000
8900 10 00000000000000000000000000000000
8a00 10 00000000000000000000000000000000
8b00 10 00000000000000000000000000000000
8c00 10 00831c0011d50a3a11000200b9000800
8d00 10 004f1c000008080007000200070108 00
c800 01 00
e803 04 00000000
e903 04 00000000

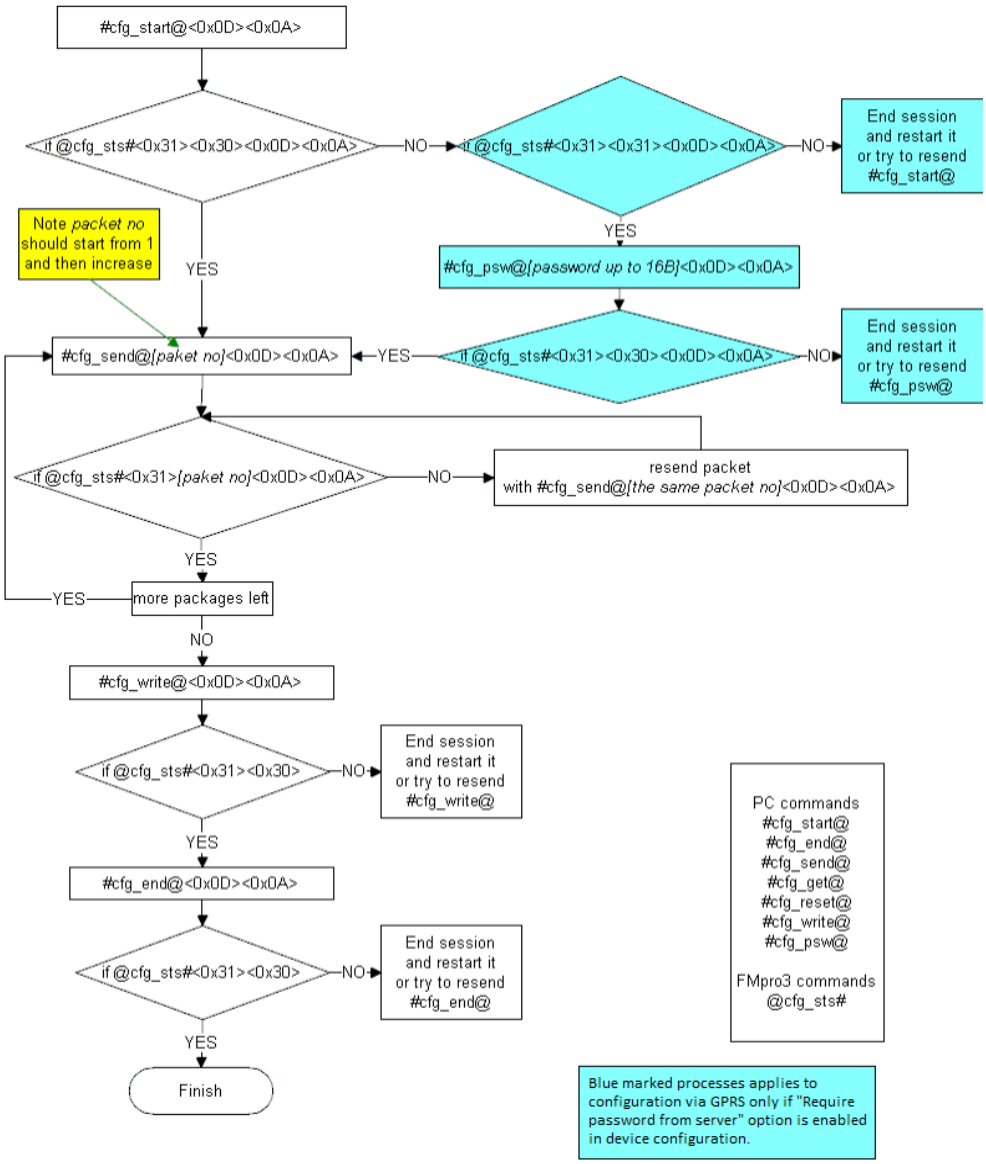
ea03 04 00000000
eb03 04 00000000
ec03 04 00000000
ed03 04 00000000
ee03 04 00000000
ef03 04 00000000
f003 04 00000000

5.2 *Configuration upload process*

Configuration process is control by 7 commands (see table below). Whole process is shown in a flowchart. At the end of this section there is an example for better understanding.

Command	Description
Host commands (server of PC)	
#cfg_start@	Enter into configuration mode (FM device).
#cfg_psw@	Enter password (only for configuration via serial cable).
#cfg_send@	Send configuration data packet to FM device.
#cfg_get@	Get configuration data packet from FM device.
#cfg_write@	Start writing configuration (recently sent) to FM device memory.
#cfg_end@	Exit configuration mode (FM device).
#cfg_reset@	Clear recently sent configuration data packets.
FM device commands	
@cfg_sts#	Report FM device status about last received command. 10 – OK, 1<*hex_number*> - OK with packet number, 11 – ERROR, 01 – ERROR (config password is incorrect).
@cfg#	Send configuration data packet from FM device to server or PC.

Configuration upload flowchart



1. Start configuration process by sending command *#cfg_start@*. Positive response from device - *@cfg_sts#10.*

2. Start sending data packets with packet number parameter: *#cfg_send@*. Positive answer - *@cfg_sts#1<0x01>*.

3. Do step 2 until whole configuration packers are sent.

4. After all configuration packets are sent server should send command to write new configuration to FM device memory: #cfg_write@. Positive answer from device: *@cfg_sts#10.* Negative answer from device: *@cfg_sts#00* (wrong configuration).

5 After successful write operation server should send exit from configuration mode command: #cfg_end@. Positive answer from FM device: *@cfg_sts#10.*

Configuration operations via GPRS are done using commands 2/102 (see 2.2.2).

Example of uploading configuration from server to FM device. Protocol which is in 2.2.2 is omitted. Just payload data (see 2.2.2) is shown.

Remarks:

<> - hex value
Data send format: *#cfg_send@<config_data_packet><0x0D><0x0A>*
Acknowledgment format: *@cfg_sts#<0x31><config_packet_number>*

Source	Command	Hex string	Description
Server	#cfg_start@<0x0D><0x0A>	23 63 66 67 5F 73 74 61 72 74 40 0D 0A	Start configuration mode.
Device	@cfg_sts#10<0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 **31 30** 0D 0A	Mode changed successfully. Status: **10**.
Server	#cfg_send@<data><0x0D><0x0A>	23 63 66 67 5F 73 65 6E 64 40 **00 02** 01 24 00 64 00 01 00 65 00 07 76 ... 0D 0A	Send packet 1 data to FM device. Packet size **512** bytes.
Device	@cfg_sts#<0x31><0x01><0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 31 01 0D 0A	Confirm that packet 1 is received.
Server	#cfg_send@<data><0x0D><0x0A>	23 63 66 67 5F 73 65 6E 64 40 **00 02** 02 4b 00 97 01 01 00 98 ... 0D 0A	Send packet 2 data to FM device. Packet size **512** bytes.
Device	@cfg_sts#<0x31><0x02><0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 31 02 0D 0A	Confirm that packet 2 is received.
...	...		
Server	#cfg_send@<data><0x0D><0x0A>	23 63 66 67 5F 73 65 6E 64 40 **FD 01** 12 48 a8 10 04 00 00 00 00 ... 0D 0A	Send packet 18 data to FM device. Packet size **509** bytes. The las packet was sent.
Device	@cfg_sts#<0x31><0x12><0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 31 12 0D 0A	The last packet (18) was received.
Server	#cfg_write@<0x0D><0x0A>	23 63 66 67 5f 77 72 69 74 65 40 0D 0A	Write configuration to FM device memory.
Device	@cfg_sts#10<0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 **31 30** 0D 0A	Config was written successfully. Status:**10**
Server	#cfg_end@<0x0D><0x0A>	23 63 66 67 5F 65 6E 64 40 0D 0A	Exit configuration mode.
Device	@cfg_sts#10<0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 **31 30** 0D 0A	Mode changed successfully. Status: **10**

5.3 *Configuration download from FM device*

Configuration download process is quite similar to configuration upload. Example of downloading configuration from FM device is shown below.

Remarks:
<> - hex value
Format data request: *#cfg_get@<packet number><0x0D><0x0A>*
Format data send: *@cfg#<config_data_packet>*

Source	Command	Hex string	Description
Server	#cfg_start@<0x0D><0x0A>	23 63 66 67 5F 73 74 61 72 74 40 0D 0A	Start configuration session.
Device	@cfg_sts#10<0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 **31 30** 0D 0A	Response with status **10.**
Server	#cfg_get@<0x01><0x0D><0x0A>	23 63 66 67 5F 67 65 74 40 **01** 0D 0A	Try to get packet with number **1.**
Device	@cfg#<data>	40 63 66 67 23 **00 02** 01 24 00 64 00 01 00 65 00 07 76 ...	Send packet 1. Packet size **512** bytes.
Server	#cfg_get@<0x02><0x0D><0x0A>	23 63 66 67 5F 67 65 74 40 **02** 0D 0A	Try to get packet with number **2.**
Device	@cfg#<data>	40 63 66 67 23 **00 02** 02 4b 00 97 01 01 00 98 ..	Send packet 2. Packet size **512** bytes.
...	...		
Server	#cfg_get@<0x15><0x0D><0x0A>	23 63 66 67 5F 67 65 74 40 **15** 0D 0A	Try to get packet with number **15.**
Device	@cfg#<0x00><0x00><0x15><0x0D><0x0A>	40 63 66 67 23 **00 00** 15 0D 0A	Packet length = **0**, last packet = 15. This means the end of get configuration operation.
Server	#cfg_end@<0x0D><0x0A>	23 63 66 67 5F 65 6E 64 40 0D 0A	End configuration session.
Device	@cfg_sts#10<0x0D><0x0A>	40 63 66 67 5F 73 74 73 23 **31 30** 0D 0A	Response with status **10**.

6 Firmware

There are two possible ways to write firmware (FW) to FM device. One option is to use USB connection with PC (personal computer). Another option is to use GPRS service (air). There are specific rules how to upload new firmware. Firmware file has extensions *.efwt* (Tco), *.efwp* (Pro) and *.efwe* (Eco). You cannot write different type of firmware to specific device, for ex.: Tco firmware to Pro FM device. Also downgrade action for some older versions of firmware is prohibited. Usage of newest firmaware is adviced. For FW update via GPRS commands 4/104 (see 2.2.4) are used.

6.1 *Firmware data packet*

Firmware file should be sent to FM device in data packets. At first you should divide FW file (raw data) into pieces of maximum 512 bytes. After this FU (Firmware Update) packets can be formed.

FU packet:

FU packet			
Whole FU packet length [2B] big endian.	**FU packet ID [2B]** big endian. Starts from no. 1.	**FU raw data [max 512B].** From *.efw* file.	**FU data CRC [2B]** big endian. Only from FU raw data.

Parameters description:

Parameter	Description
Whole FU packet length	Length of current FU packet size. Whole FU packet length = length + ID + raw data + crc = 6 + raw data. Format: big endian. Parameter size is 2 bytes.
FU packet ID	Unique FU packet ID. This parameter identifies packets. Format: big endian. Parameter size is 2 bytes.
FU raw data	Raw data of part of firmware file. Max size is 512 bytes.
FU data CRC	CRC16 kermit calculation. It is calculated only from FU raw data. Format: big endian. Parameter size is 2 bytes.

Example of FU data packet (raw data 518 bytes):

06020100D732356607162437330130 6FE0656754220E073311333801515F5140687A4040784E00647200245E7937017566677245771300350177 7A01D5507803712D522A422F357673352637E1FF306F941A645446AB0733D1E338012D885140D5CE404067FB00641163245E04DA017560667 2454913003562147A0143517803532D522A042A3576291624370101306F4AF36754A8730433518F380169A451400B19404081F00264A5D3245 E7E5901756066724549130035 8C187A0143517803532D522A042A35762916243701013 06F6DF065AC779D81CB4CF0A8E90830D304EB3EEA B17949DA2173D124AE02CFAE870A0BC8AD257013C5007862BEDB283BF3314D4A6D9BCD367676F0273768267F9F6D0C77AC765614C34 15430BE18C4500B756B48FF68B6013F161E218E69CF001E671B73BD2B1BF9E41287720E3E8D68FB3105FF366B523CA7276B7C88639A31A 497D562B46395065847A8396A6522A895F9389693080900647223247A79120153133F5AFA52B1FBED537052BE10F330BC3B2E226D0160028 6E4F5275F796818D02CF0E755134435C3BDD54D283ACCA9C5105372B013C8E44D40F008D81A1E3285FB86016C1980FDB601570A463F87 4EF3EFB161DA00B531764A55241657923A902D1C0734774D071273A0D6FFE85C4EFD78CF50FD4BBE7C996346DBA99EC80185BDFC41B 5808D0381FE8088FE238E4BF39BB3522A1985

Whole FU packet length = 0x06 0x02 (raw) = 0x0206 (Endian conversation) = 518 bytes.

FU packet ID = 0x01 0x00 (raw) = 0x0001 (Endian conversation) = 1

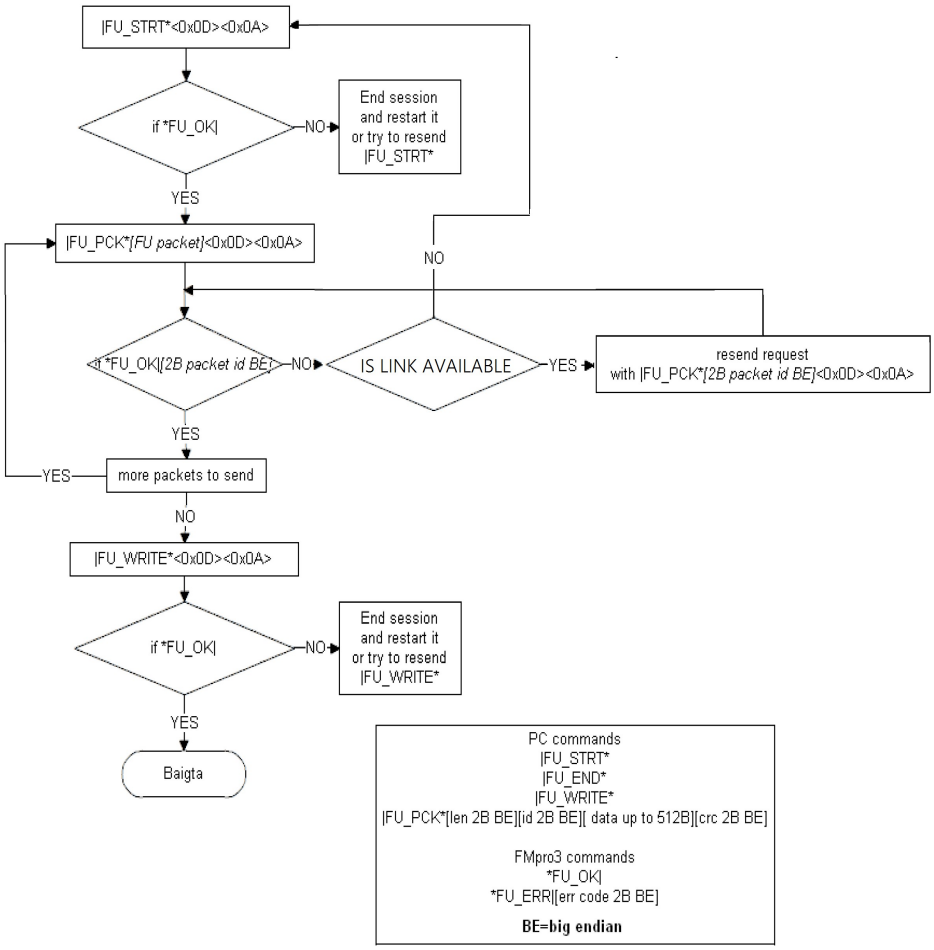
FU data CRC = 0x19 0x85 (raw) = 0x8519 (Endian) = 34073

6.2 *Firmware upload process*

Special commands are used for FW update. All commands are terminated with <0x0D> <0x0A> symbols.

Command	Description
Host commands (server of PC)	
|FU_STRT*	Enter into firmware update mode (FM device).
|FU_PCK*	Send firmware data packet.
|FU_WRITE*	Start writing firmware (recently sent) to FM device memory.
|FU_END*	Terminate firmware update. Only used to quit FW update process.
FM device commands	
*FU_OK|	Acknowledgment from FM device.
*FU_EER|	Error from FM device. Wrong CRC result is shown if necessary (last 2 bytes).

Firmware update flowchart



1. Enter to FW update mode */FU_STRT<0x0D><0x0A>*. FM device should respond with **FU_OK|<0x0D><0x0A>*.
2. Send FW packet with */FU_PCK<FU_packet><0x0D><0x0A>.* FM device should respond with **FU_OK|<FU packet ID>.*
3. Repeat step 2 until all FU packets are sent. If **FU_OK|* isn't received check for the GPRS link availability and if link available repeat packet, if not available – reconnect and start from beginning (*/FU_STRT**).
4. Start writing FW to FM device by sending command: */FU_WRITE*<0x0D><0x0A>*. FM device reconstructs firmware file from received data packets. FM device should respond: **FU_OK|<0x0D><0x0A>*. After that FM device automatically ends FW update mode.

FW update operations via GPRS are done using commands 4/104 (see 2.2.4).

Example of uploading firmware from server to FM device. Server protocol is omitted. Just payload data (see 2.2.4) is shown.

Source	Command	Hex string	Description
Server	|FU_STRT*<0x0D><0x0A>	7C 46 55 5F 53 54 52 54 2A 0D 0A	Enter FW update mode.
Device	*FU_OK|<0x0D><0x0A>	2A 46 55 5F 4F 4B 7C 0D 0A	Mode changed successfully.
Server	|FU_PCK*<FW data packet><0x0D><0x0A>	7C 46 55 5F 50 43 4B 2A **06 02** 01 00 D7 ... 19 85 0D 0A	Send FW data packet 1 to FM device. Packet size **518** bytes.
Device	*FU_OK|<0x0D><0x0A>	2A 46 55 5F 4F 4B 7C 01 00 0D 0A	Confirm that packet 1 is received.
Server	|FU_PCK*<FW data packet><0x0D><0x0A>	7C 46 55 5F 50 43 4B 2A **06 02** 02 00 E7 … 2E FF 0D 0A	Send FW data packet 2 to FM device. Packet size **518** bytes.
Device	*FU_OK|<0x0D><0x0A>	2A 46 55 5F 4F 4B 7C 02 00 0D 0A	Confirm that packet 2 is received.
...	...		
Server	|FU_PCK*<FW data packet><0x0D><0x0A>	7C 46 55 5F 50 43 4B 2A **06 02** C5 01 E7 … 69 74 0D 0A	Send packet 453 data to FM device. Packet size **518** bytes. The las packet was sent.
Device	*FU_OK|<0x0D><0x0A>	2A 46 55 5F 4F 4B 7C C5 01 0D 0A	The last packet (453) was received.
Server	|FU_WRITE*<0x0D><0x0A>	7C 46 55 5F 57 52 49 54 45 2A 0D 0A	Write firmware to FM device memory.
Device	*FU_OK|<0x0D><0x0A>	2A 46 55 5F 4F 4B 7C 0D 0A	Firmware was updated successfully.

Example of whole server command 104 (see 2.2.4) hex string (with explanation) is shown below:

0211687c46555f50434b2a06020100 7f703d56071624373301306f02656754220e073311333801515f5140687a4040784e00647200245e7937017566 67724577130035017 77a01d5507803712d522a602f3576539b24370df3306f041b6554a2d407333ff7380119945140c1d240400de700641163245e1a 5401756066724549 13003562147a0143517803532d522a042a3576990024370101306f0e636754c87605334182380125b051400b19404031d40264 79c7245e885a0175606672454913003562147a0143517803532d522a042a3576291624370101306f6df065ac779d81cb4cf0a8e90830d304eb3eeab1 7949da2173d124ae02cfae870a0bc8ad257013c5007862bedb283bf3314d4a6d3b8e3776d6b2263768267f9f6d0c77ac765614c3415430be18c4500b 756b48ff68b6013f161e218e69cf001e671b73bd2b1bf9e41287720e3e8d68fb3105ff366b523ca7276b7c88639a31a497d562b46395065847a8396a6 522a895f9389693080900647223247a79120153133f5afa52b1fbed537052be10f330bc3b2e226d01601f86acf2275f796818d02cf0e755134421c359d 64d282ecc4dc6105366b0f7c8e44d54f074d81a1e2685d786016c0d80d9b601570a463f8752f3bdb175da24b531764a5524164b9206902d1c0734774 d07126fa022fee85c4efd78cf50fd5fbe58996346dba99ec80185acfc55b5ac8d0381fe8088fe238e5ff3b7b3522aeb450d0a7d4d

Packet length = 0x0211 = 529 bytes (itself and CRC16 field is excluded).

Server protocol Command ID = 0x68 = 104

Firmware packet send command = 7c46555f50434b2a = |FU_PCK*

FU packet length = 0x206 = 518 bytes.

FU packet ID = 0x0001 = 1.

Firmware raw data

Symbols to terminate payload packet - 0d0a

CRC16 Kermit answer = 0x4d7d = 19837

7 Abbreviations

<0x00> - value in hex format

<0x0D> - (CR) Carriage Return symbol

<0x0A> - (LF) Line Feed symbol. Also known as New Line (NL) / End of Line (EOL)

B – bytes/bytes

kB – kilobyte

CFG – Configuration

Config - Configuration

FW – firmware

FU – Firmware Update