# FMA120 Protocols
### V0.4

# 1.    FMA110 DATA PROTOCOL

## 1.1    AVL data array

Because the smallest information amount that can be written is one bit, there can be some bits left unused when result is byte array. Any unused bits should be left blank.

| Codec ID | Number of Data | Data | Number of Data |
|----------|----------------|------|----------------|
| 1 Byte   | 1 Byte         | …    | 1 Byte         |

Number of data – number of encoded data (number of records)
In FMA120 codec ID is 08

## 1.2    Data

| AVL Data | … | AVL Data |
|----------|---|----------|

AVL data – encoded data element

## 1.3    AVL Data

| Timestamp | Priority | GPS Element | IO Element |
|-----------|----------|-------------|------------|
| 8 Bytes   | 1 Byte   | 15 Bytes    | ...        |

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

## 1.4     Priority

| 0 | Low      |
|---|----------|
| 1 | High     |
| 2 | Panic    |
| 3 | Security |

## 1.5    GPS Element

| Longitude | Latitude | Altitude | Angle   | Satellites | Speed   |
|-----------|----------|----------|---------|------------|---------|
| 4 Bytes   | 4 Bytes  | 2 Bytes  | 2 Bytes | 1 Byte     | 2 Bytes |

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

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

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

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

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

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

## 1.6     IO element

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

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

---

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

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

| Permanent I/O elements (are always sent to server if enabled) | | | | |
|---|---|---|---|---|
| **Property ID** | **Property Name** | **Bytes** | **Description** | **Type[1]** |
| 1 | Digital Input Status 1 | 1 | Logic: 0 / 1 | M |
| 2 | Digital Input Status 2 | 1 | Logic: 0 / 1 | M |
| 3 | Digital Input Status 3 | 1 | Logic: 0 / 1 | M |
| 9 | Analog Input 1 | 2 | Voltage: mV, 0 – 30 V | M |
| 16 | Total Distance | 4 | Total distance: m | M |
| 21 | GSM level | 1 | GSM signal level value in scale 1 – 5 | M |
| 24 | Speed | 2 | Value in km/h, 0 – xxx km/h | M |
| 66 | External Power Voltage | 2 | Voltage: mV, 0 – 30 V | M |
| 67 | Battery Voltage | 2 | Voltage: mV, 0 – 30 V | O |
| 68 | Battery Current | 2 | Current, mA | O |
| 69 | GPS Status | 1 | States: 0 – GPS module is turned off, 2 – working, but no fix, 3 – working with GPS fix, 4 – GPS module is in sleep state, 5 – antenna is short circuit | M |
| 71 | Dallas Temperature ID 4 | 8 | Dallas sensor ID number | M |
| 72 | Dallas Temperature | 4 | 10 * Degrees ( °C ), -55 - +115, if 3000 – Dallas error | M |
| 73 | Dallas Temperature 2 | 4 | 10 * Degrees ( °C ), -55 - +115, if 3000 – Dallas error | M |
| 74 | Dallas Temperature 3 | 4 | 10 * Degrees ( °C ), -55 - +115, if 3000 – Dallas error | M |
| 75 | Dallas Temperature 4 | 4 | 10 * Degrees ( °C ), -55 - +115, if 3000 – Dallas error | M |
| 76 | Dallas Temperature ID 1 | 8 | Dallas sensor ID number | M |
| 77 | Dallas Temperature ID 2 | 8 | Dallas sensor ID number | M |
| 78 | iButton ID | 8 | iButton ID number | M |
| 79 | Dallas Temperature ID3 | 8 | Dallas sensor ID number | M |
| 80 | Data Mode | 1 | 0 – home on stop, 1 – home on move, 2 – roaming on stop, 3 – roaming on move, 4 – unknown on stop, 5 – unknown on move | M |
| 179 | Digital Output 1 state | 1 | Logic: 0 / 1 | M |
| 180 | Digital Output 2 | 1 | Logic: 0 / 1 | M |

| | | | | |
|---|---|---|---|---|
| | state | | | |
| 181 | PDOP | 2 | Probability * 10; 0-500 | M |
| 182 | HDOP | 2 | Probability * 10; 0-500 | M |
| 199 | Odometer Value (Virtual Odometer) | 4 | Distance between two records: m | M |
| 200 | Deep Sleep | 1 | 0 – not deep sleep mode, 1 – deep sleep mode | M |
| 205 | Cell ID | 2 | GSM base station ID | M |
| 206 | Area Code | 2 | Location Area code (LAC), it depends on GSM operator. It provides unique number which assigned to a set of base GSM stations. Max value: 65536 | M |
| 239 | Ignition | 1 | 0 – ignition off, 1 – ignition on | M |
| 240 | Movement Sensor | 1 | 0 – not moving, 1 – moving | M |
| 241 | GSM Operator Code | 4 | Currently used GSM Operator code | M |
| **LVCAN I/O elements** **(are sent to server if configured)** | | | | |
| Property ID | Property Name | Bytes | Description | Type |
| 81 | LVCAN Speed | 1 | Value in km/h | A[2] |
| 82 | LVCAN Accelerator Pedal Position | 1 | Value in persentages, % | A[2] |
| 83 | LVCAN Total Fuel Used | 4 | Value in liters multiplied by 10, L*10 | A[2] |
| 84 | LVCAN Fuel Level (liters) | 2 | Value in liters, L | A[2] |
| 85 | LVCAN Engine RPM | 2 | Value in rounds per minute, rpm | A[2] |
| 87 | LVCAN Vehicle Distance | 4 | Value in meters, m | A[2] |
| 89 | LVCAN Fuel Level (percentage) | 1 | Value in percentages, % | A[2] |
| 90 | LVCAN Door Status | 2 | Door status value: Min – 0, Max – 16128<br>Door status is represented as bitmask converted to decimal value. Possible values:<br>0 – all doors closed,<br>0x100 (256) – front left door is opened,<br>0x200 (512) – front right door is opened,<br>0x400 (1024) – rear left door is opened,<br>0x800 (2048) – rear right door is opened,<br>0x1000 (4096) – hood is opened,<br>0x2000 (8192) – trunk is opened,<br>0x3F00 (16128) – all doors are opened,<br>or combinations of values | A[2] |
| 100 | LVCAN Program Number | 4 | Value: Min – 0, Max – 999 | A[2] |
| 101 | LVC ModuleID | 8 | Module ID | A[2] |
| 102 | LVC Engine Work Time | 4 | Engine work time in minutes | A[2] |
| 103 | LVC Engine Work Time (counted) | 4 | Total Engine work time in minutes | A[2] |

| 105 | LVC Total Mileage (counted) | 4 | Total Vehicle Mileage, m | $A^2$ |
|---|---|---|---|---|
| 107 | LVC Fuel Consumed (counted) | 4 | Total Fuel Consumed,liters * 10 | $A^2$ |
| 110 | LVC Fuel Rate | 2 | Fuel Rata, liters *10 | $A^2$ |
| 111 | LVC AdBlue Level (percent) | 1 | AdBlue, % | $A^2$ |
| 112 | LVC AdBlue Level (liters) | 2 | AdBlue level, L | $A^2$ |
| 114 | LVC Engine Load | 1 | Engine load, % | $A^2$ |
| 115 | LVC Engine Temperature | 2 | Engine Temperature, 10 * Degrees ( °C ), | $A^2$ |
| 118 | LVC Axle 1 Load | 2 | Axle 1 load, kg | $A^2$ |
| 119 | LVC Axle 2 Load | 2 | Axle 2 load, kg | $A^2$ |
| 120 | LVC Axle 3 Load | 2 | Axle 3 load, kg | $A^2$ |
| 121 | LVC Axle 4 Load | 2 | Axle 4 load, kg | $A^2$ |
| 122 | LVC Axle 5 Load | 2 | Axle 5 load, kg | $A^2$ |
| 123 | LVC Control State Flags | 4 | Control state flags | $A^2$ |
| 124 | LVC Agricultural Machinery Flags | 8 | Agricultural machinery flags | $A^2$ |
| 125 | LVC Harvesting Time | 4 | Harvesting Time, minutes | $A^2$ |
| 126 | LVC Area of Harvest | 4 | Area of Harvest, m^2 | $A^2$ |
| 127 | LVC Mowing Efficiency | 4 | Mowing efficiency, (m^2)/h | $A^2$ |
| 128 | LVC Grain Mown Volume | 4 | Mown Volume, kg | $A^2$ |
| 129 | LVC Grain Moisture | 2 | Grain Moisture in proc, % | $A^2$ |
| 130 | LVC Harvesting Drum RPM | 2 | Harvesting Drum RPM, RPM | $A^2$ |
| 131 | LVC Gap Under Harvesting Drum | 1 | Gap Under Harvesting Drum, mm | $A^2$ |
| 132 | LVC Security State Flags | 8 | Security State Flag | $A^2$ |
| 133 | LVC Tacho Total Vehicle Distance | 4 | Tacho Total Vehicle Distance, m | $A^2$ |
| 134 | LVC Trip Distance | 4 | Trip Distance, m | $A^2$ |
| 135 | LVC Tacho Vehicle Speed | 2 | Tacho Vehicle Speed, km/h | $A^2$ |
| 136 | LVC Tacho Driver Card Presence | 1 | Tacho Driver Card Presence | $A^2$ |
| 137 | LVC Driver1 States | 1 | Driver1 States | $A^2$ |
| 138 | LVC Driver2 States | 1 | Driver2 States | $A^2$ |
| 139 | LVC Driver1 Continuous Driving Time | 2 | Driver1 Continuous Driving Time, minutes | $A^2$ |

| Property ID | Property Name | Bytes | Description | Type |
|---|---|---|---|---|
| 140 | LVC Driver2 Continuous Driving Time | 2 | Driver2 Continuous Driving Time, minutes | $A^2$ |
| 141 | LVC Driver1 Cumulative Break Time | 2 | Driver1 Cumulative Break Time, minutes | $A^2$ |
| 142 | LVC Driver2 Cumulative Break Time | 2 | Driver2 Cumulative Break Time, minutes | $A^2$ |
| 143 | LVC Driver1 Duration Of Selected Activity | 2 | Driver1 Duration Of Selected Activity, minutes | $A^2$ |
| 144 | LVC Driver2 Duration Of Selected Activity | 2 | Driver2 Duration Of Selected Activity, minutes | $A^2$ |
| 145 | LVC Driver1 Cumulative Driving Time | 2 | Driver1 Cumulative Driving Time, minutes | $A^2$ |
| 146 | LVC Driver2 Cumulative Driving Time | 2 | Driver2 Cumulative Driving Time, minutes | $A^2$ |
| 147 | LVC Driver1 ID High | 8 | Driver1 ID High | $A^2$ |
| 148 | LVC Driver1 ID Low | 8 | Driver1 ID Low | $A^2$ |
| 149 | LVC Driver2 ID High | 8 | Driver2 ID High | $A^2$ |
| 150 | LVC Driver2 ID Low | 8 | Driver2 ID Low | $A^2$ |
| 151 | LVC Battery Temperature | 2 | 10* Degrees, ( °C ) | $A^2$ |
| 152 | LVC Battery Level (percent) | 1 | Value in percentages, % | $A^2$ |
| **Eventual I/O elements** <br> **(generate and send record to server only if appropriate conditions are met)** | | | | |
| **Property ID** | **Property Name** | **Bytes** | **Description** | **Type** |
| 155 | Geofence zone 01 | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 156 | Geofence zone 02 | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 157 | Geofence zone 03 | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 158 | Geofence zone 04 | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 159 | Geofence zone 05 | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 175 | Auto Geofence | 1 | Event: 0 – target left zone, 1 – target entered zone | ME |
| 249 | Jamming | 1 | 1 – jamming start, 0 – jamming stop | ME |
| 250 | Trip | 1 | 1 – trip start, 0 – trip stop | ME |
| 251 | Immobilizer | 1 | 0 – iButton not connected, 1 – iButton connected (Immobilizer), 2 – iButton connected (Authorized Driving) | ME |

| 253 | Green driving type | 1 | 1 – harsh acceleration, 2 – harsh braking, 3 – harsh cornering | ME |
|---|---|---|---|---|
| 254 | Green driving value | 1 | Depending on green driving type: if harsh acceleration or braking – g*100 (value 123 -> 1.23g), if harsh cornering – degrees (value in radians) | ME |
| 255 | Over Speeding | 1 | At over speeding start km/h, at over speeding end km/h | ME |

[1] M – mandatory (available on all hardware revisions), A – additional hardware required, O – optional (hardware dependent), E – event only.

## 1.7 Example

```
Received data:
080400000113fc208dff000f14f650209cca80006f00d60400040004030101150316030000
1460000015d0000000113fc17610b000f14ffe0209cc580006e00c00500010000403010115
0316010001460000015e0000000113fc284945000f150f00209cd20000950108040000000
403010115001603000014600000015d0000000113fc267c5b000f150a50209cccc000930068
04000000004030101150016030001460000015b0004
```

**08** – Codec ID
    **04**- Number of Data (4 records)

       **1'st record data**

        **00000113fc208dff** – Timestamp in milliseconds (1185345998335 → 1185345998,335 in Unix Timestamp = 25 Jul 2007 06:46:38 UTC)
        **00** – Priority

        **GPS Element**

        **0f14f650** – Longitude 253032016 = 25,3032016° N
        **209cca80** – Latitude 547146368 = 54,7146368 ° E
        **006f** – Altitude 111 meters
        **00d6** – Angle 214°
        **04** – 4 Visible sattelites
        **0004** – 4 km/h speed

        **IO Element**

        **00** – IO element ID of Event generated (in this case when 00 – data generated not on event)
        **04** – 4 IO elements in record
        **03** – 3 IO elements, which length is 1 Byte
        **01** – IO element ID = 01
        **01** – 1'st IO element's value = 1
        **15** – IO element ID = 21
        **03** – 21'st IO element's value = 3
        **16** – IO element ID = 22
        **03** – 22'nd IO element's value = 3

**00** – 0 IO elements, which value length is 2 Bytes
**01** – 1 IO element, which value length is 4 Bytes
**46** – IO element ID = 70
**0000015d** – 70'th IO element's value = 349
**00** – 0 IO elements, which value length is 8 Bytes

**2'nd record data**

    00000113fc17610b 00 0f14ffe0209cc580006e00c7050001
0004030101150316010001460000015e00

**3'd record data**

    00000113fc284945 00 0f150f00209cd20000950108040000
0004030101150016030001460000015d00

**4'th record data**

    00000113fc267c5b 00 0f150a50209cccc000930068040000
0004030101150016030001460000015b00


**04** – Number of Data (4 records)

# 2.    SENDING DATA OVER TCP/IP

## 2.1    AVL data packet

AVL packet is used to encapsulate AVL data and send it to server.

| Four zeros | Data length | Data | Crc |
|------------|-------------|------|-----|

| | |
|---|---|
| Four zeros | Four zero bytes (0x00) |
| Data length | Number of bytes in data field (Integer) |
| Data | Any AVL data array |
| CRC | 16bit CRC value of data (Integer). Polynomial 0xA001. |

## 2.2    Communication with server

First when module connects to server, module sends its IMEI. IMEI is sent the same way as encoding barcode. First comes short identifying number of bytes written and then goes IMEI as text (bytes).

For example IMEI 123456789012345 would be sent as 000F313233343536373839303132333435

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

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

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

Example:
Module connects to server and sends IMEI:
000F313233343536373839303132333435
Server accepts the module:
01
Module sends data packet:

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

Server acknowledges data reception (2 data elements):
00000002

# 3. SENDING DATA OVER UDP/IP

## 3.1 UDP channel protocol

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

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

| Packet Type | |
|---|---|
| 1 | Data packet requiring acknowledgment |

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

| Acknowledgment packet | | |
|---|---|---|
| Packet length | 2 bytes | 0x0003 |
| Packet id | 2 bytes | same as in acknowledged packet |
| Packet type | 1 byte | 0x01 |

## 3.2 Sending AVL data using UDP channel

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

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

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

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

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

Scenario:
Module sends UDP channel packet with encapsulated AVL data packet (*Packet* type=1).
Server sends UDP channel packet with encapsulated response (*Packet type*=1)
Module validates *AVL packet id* and *Number of accepted AVL elements*. If server response with valid *AVL packet id* is not received within configured timeout, module can retry sending.

Example:
Module sends the data:

| UDP channel header | AVL packet header | AVL data array |
|---|---|---|
| Len – 253, Id – 0xCAFE, Packet type – 01 | AVL packet id – 0xDD, IMEI – 1234567890123456 | CodecId – 08, NumberOfData – 2. (Encoded using continuous bit stream) |
| 00FDCAFE01 | DD000F31333435363738393031323334 35 | 0802…(data elements)…02 |

Server must respond with acknowledgment:

| UDP channel header | AVL packet acknowledgment |
|---|---|
| Len – 5, Id – 0xCAFE, Packet type – 01 | AVL packet id – 0xDD, NumberOfAcceptedData – 2 |
| 0005CAFE01 | DD02 |

# 4. SENDING DATA USING SMS

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

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

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

# 5. 24 POSITION SMS DATA PROTOCOL

24-hour SMS is usually sent once every day and contains GPS data of last 24 hours. TP-DCS field of this SMS should indicate that message contains 8-bit data (i.e. TP-DCS can be 0x04).

Note, that 24 position data protocol is used only with subscribed SMS. Event SMS use standard AVL data protocol.

## 5.1 Encoding

To be able to compress 24 GPS data entries into one SMS (140 octets), the data is encoded extensively using bit fields. Data packet can be interpreted as a bit stream, where all bits are numbered as follows:

| Byte 1 | Byte 2 | Byte 3 | Bytes 4-… |
|--------|--------|--------|-----------|
| Bits 0-7 | Bits 8-15 | Bits 16-24 | Bits 25-… |

Bits in a byte are numbered starting from least significant bit. A field of 25 bits would consist of bits 0 to 24 where 0 is the least significant bit and bit 24 – most significant bit.

## 5.2 Structure

| SMS Data Structure | | | |
|---|---|---|---|
| | Size (bits) | Field | Description |
| | 8 | CodecId | CodecId = 4 |
| | 35 | Timestamp | Time corresponding to the first (oldest) GPS data element, represented in seconds elapsed from 2000.01.01 00:00 EET. |
| | 5 | ElementCount | Number of GPS data elements. |
| ElementCount * | | GPSDataElement | GPS data elements. |
| | | Byte-aling padding | Padding bits to align to 8-bits boundary |
| | 64 | IMEI | IMEI of sending device as 8-byte long integer |

The time of only the first GPS data element is specified in *Timestamp* field. Time corresponding to each further element can be computed as e*lementTime = Timestamp* + (1 hour * elementNumber).

| GPSDataElement | | | | |
|---|---|---|---|---|
| | | Size (bits) | Field | Description |
| | | 1 | ValidElement | ValidElement=1 – there is a valid GpdDataElement following, ValidElement=0 – no element at this position. |

| GPSDataElement | | | | | |
|---|---|---|---|---|---|
| ValidElement == 1 | | | 1 | DifferentialCoords | Format of following data. |
| | DifferentialCoords == 1 | | 14 | LongitudeDiff | Difference from previous element's longitude. LongitudeDiff = prevLongitude – Longitude + $2^{13}$ – 1 |
| | | | 14 | LatitudeDiff | Difference from previous element's latitude LatitudeDiff = prevLatitude – Latitude + $2^{13}$ – 1 |
| | DifferentialCoords == 0 | | 21 | Longitude | Longitude= {(LongDegMult + 18 * $10^8$) * ($2^{21}$ – 1)} over {36*$10^8$} |
| | | | 20 | Latitude | Latitude=(LatDegMult + 9*$10^8$) * ($2^{20}$ – 1) over {18*$10^8$} |
| | | | 8 | Speed | Speed in km/h. |

| | |
|---|---|
| *Longitude* | longitude field value of *GPSDataElement* |
| *Latitude* | latitude field value of *GPSDataElement* |
| *LongDegMult* | longitude in degrees multiplied by $10^7$ (integer part) |
| *LatDegMult* | latitude in degrees multiplied by $10^7$ (integer part) |
| *prevLongitude* | longitude field value of previous *GPSDataElemen* |
| *prevLatitude* | latitude field value of previous *GPSDataElement* |

## 5.3   Decoding GPS position

When decoding GPS data with *DifferentialCoords*=1, *Latitude* and *Longitude* values can be computed as follows: *Longitude*=*prevLongitude* – *LongitudeDiff* + $2^{13}$ – 1, Latitude=*prevLatitude* – *LatitudeDiff* + $2^{13}$ – 1.

If there were no previous non-differential positions, differential coordinates should be computed assuming prevLongitude=prevLatitude=0.

When *Longitude* and *Latitude* values are known, longitude and latitude representation in degrees can be computed as follows:

$$LongDeg = \frac{Longitude * 360}{2^{21} - 1} - 180 \qquad LatDeg = \frac{Latitude * 180}{2^{20} - 1} - 90$$

# 6. CHANGE LOG

02651

| Nr. | Date | New version number | Comments |
|-----|------|--------------------|----------|
| 1 | 161012 | 0.1 | File created |
| 2 | 161012 | 0.1 | Table of I/O Elements updated to correspond 01.21.xx branch |
| 3 | 170225 | 0.2 | Table of I/O Elements updated with LVCAN Door Status element |
| 4 | 170228 | 0.3 | Table of I/O Elements updated |
| 5 | 170329 | 0.4 | Dallas Temperature ID 1/2/3/4 property ID changes |