Cellocator Wireless Communication Protocol - CR300

Proprietary and Confidential

Version 4.4.1.3

Revised and Updated: November 7, 2019

POINTER TELOCATION LTD. 14 HAMELACHA ST. ROSH HA'AYIN 4809133, ISRAEL • TEL: 972-3-5723111 • FAX: 972-3-5723100 • WWW.POINTER.COM

Legal Notices

IMPORTANT

1. All legal terms and safety and operating instructions should be read thoroughly before the product accompanying this document is installed and operated.
2. This document should be retained for future reference.
3. Attachments, accessories or peripheral devices not supplied or recommended in writing by Pointer Telocation Ltd. May be hazardous and/or may cause damage to the product and should not, in any circumstances, be used or combined with the product.

General

The product accompanying this document is not designated for and should not be used in life support appliances, devices, machines or other systems of any sort where any malfunction of the product can reasonably be expected to result in injury or death. Customers of Pointer Telocation Ltd. Using, integrating, and/or selling the product for use in such applications do so at their own risk and agree to fully indemnify Pointer Telocation Ltd. For any resulting loss or damages.

Warranty Exceptions and Disclaimers

Pointer Telocation Ltd. Shall bear no responsibility and shall have no obligation under the foregoing limited warranty for any damages resulting from normal wear and tear, the cost of obtaining substitute products, or any defect that is (i) discovered by purchaser during the warranty period but purchaser does not notify Pointer Telocation Ltd. Until after the end of the warranty period, (ii) caused by any accident, force majeure, misuse, abuse, handling or testing, improper installation or unauthorized repair or modification of the product, (iii) caused by use of any software not supplied by Pointer Telocation Ltd., or by use of the product other than in accordance with its documentation, or (iv) the result of electrostatic discharge, electrical surge, fire, flood or similar causes. Unless otherwise provided in a written agreement between the purchaser and Pointer Telocation Ltd., the purchaser shall be solely responsible for the proper configuration, testing and verification of the product prior to deployment in the field.

POINTER TELOCATION LTD.'S SOLE RESPONSIBILITY AND PURCHASER'S SOLE REMEDY UNDER THIS LIMITED WARRANTY SHALL BE TO REPAIR OR REPLACE THE PRODUCT HARDWARE, SOFTWARE OR SOFTWARE MEDIA (OR IF REPAIR OR REPLACEMENT IS NOT POSSIBLE, OBTAIN A REFUND OF THE PURCHASE PRICE) AS PROVIDED ABOVE. POINTER TELOCATION LTD. EXPRESSLY DISCLAIMS ALL OTHER WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, SATISFACTORY PERFORMANCE AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL POINTER TELOCATION LTD. BE LIABLE FOR ANY INDIRECT, SPECIAL, EXEMPLARY, INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOSS OR INTERRUPTION OF USE, DATA, REVENUES OR PROFITS) RESULTING FROM A BREACH OF THIS WARRANTY OR BASED ON ANY OTHER LEGAL THEORY, EVEN IF POINTER TELOCATION LTD. HAS BEEN ADVISED OF THE POSSIBILITY OR LIKELIHOOD OF SUCH DAMAGES.

Intellectual Property

Table of Contents

1 Introduction

1.1 About this Document

This document describes the unit wireless communication protocol structure, implemented in Cellocator units. It describes every byte of the inbound/outbound messages, which can be sent/received by the unit over the air.

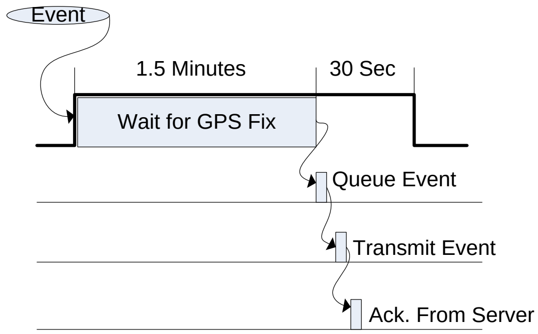
The document comprises of the following main parts:

- Telemetry Channel (outbound messages initiated from the unit towards the server)
- Command Channel (inbound messages initiated from the server towards the unit)
- CSA Channel

Most communication flow scenarios between the unit and the server implement acknowledge from the receiving side to the sending side. Some are done using generic ACK (acknowledge) message, and some are done using other messages dedicated to the specific scenario.

A large portion of the outbound messages from the unit to the server, are initiated by the unit in a response to a certain trigger (e.g. GPIO activation, speed violation, etc.). Those messages are referred as events. The unit supports the following kinds of events:

- **Logged Event:** If the condition for a specific logged event is met, the unit will create an event and store it into its non-volatile memory. The event will be sent to the server only during the GPRS session and will be deleted from the memory of the unit only after reception of acknowledge from the server. Note: Plain events will never be delivered by SMS.
- **Distress Event:** If the condition for a specific distress event is met, the unit will create a series of messages (session). The messages will be sent to the server immediately with the first available communication transport (during GPRS session – over IP, otherwise by SMS). The messages are not stored in the unit memory and if there is no cellular coverage at the moment of sending the message will be lost. Distress events do not require acknowledge from the server.
- **Active Logged Event:** This event is designed to enhance the functionality of legacy logged events. It is important for units such as CelloTrack, which are battery operated and mostly hibernating while periodically communicating with the server. Enabling the Active Logged Event feature changes the behavior of the unit in the following way:
  - During Hibernation
    When a new event is generated, the unit will turn its modem and GPS on, wait for a GPS fix and then queue the event into the event queue. The event will be transmitted to the server, acknowledged by the server and removed from the queue. Active Logged Event turns the unit on from hibernation for up to 2 minutes. If a GPS fix is not detected within 1.5 minutes from the beginning of the session, the event will be queued into the events queue and sent towards the server while giving an extra 30 seconds for the server to acknowledge the event. If a cellular link is not available the unit will be turned off and the message will wait in the queue for later delivery.



-
-
- During Live Tracking
  When a new event is generated, and the GPS is off (in CelloTrack units), the unit will turn the GPS on, wait for a fix and then insert the event into the event queue.

Naturally, the wireless protocol has evolved over the years, to answer the growing needs, and old lean message types are gradually replaced by newer message type (Type 11), which has more robust and modular structure, intended to support longer diverse messages. Thus, it is recommended to implement the complete Type 11 on the server side.

1.2 Abbreviations

Abbreviation	Description
ACK	Acknowledge
CAN	Controller Area Network
CCC	Command and Control Center
DB	Database
FMS	Fleet Management System
OTA	Over the Air
PDU	Protocol Description Unit (Common name for data SMS)
PGN	Parameter Group Number
SMS	Short Message Service (GSM)
PTR	Pointer Telocation Ltd.
PSP	Pointer Serial Protocol, normally refers to a Car Alarm System interfacing through this protocol
NVM	Non Volatile Memory
FW	Firmware
HW	Hardware

1.3 References

All the reference documents listed in the following table can be downloaded from the support section of the Pointer Website (www.pointer.com).

#	Reference	Description
1	Cellocator Programming Manual	This document describes the features supported by the Cellocator unit and provides details about the parameters of its configuration.
2	Cellocator Hardware Installation Guides	This document provides all necessary information for a technician who is involved in the installation of Cellocator units. It describes how to install and verify the proper functioning of the unit installation kit elements.
3	Cellocator Serial Communication Protocol	This document describes the serial interface (RS232) protocol

2 Telemetry Channel (Outbound Messages)

2.1 Overview

The telemetry channel comprises several types of messages, as described in the following:

- **Status/location Message (Message Type 0)** – a legacy message, which is sent by default, as a reply to a command or as the message of choice when reporting events or emergency situations.
- **Programming Data (Message Type 3)** – this message is sent as a reply to programming commands, or by request. It contains the new contents of the programmed block, which allows verification of the programming.
- **Logged Fragment of Forwarded Data from Serial Port to Wireless Channel (Message Type 7)** – this message is sent when the terminal, connected to the serial port of the unit is forwarding data to the central control through unit log.
- **Real Time Forwarded Data from Serial Port to Wireless Channel (Message Type 8)** – this legacy message is sent when the terminal, connected to the serial port of the unit is forwarding data to the central control without logging it.
- **Modular Message (Message Type 9)** – this legacy modular message is designed to contain different types of data, such as CAN bus sensors, Cell ID, debug data, etc.
- **Modular Message (Message Type 11)** – this modular message type implements an extended modular protocol, intended to replace older message types (0, 3, and 9). It is currently used for CAN bus applications, CelloTrack Nano, CelloTrack-4 family, configuration memory programming and uploading of devices with 8 Kbytes of configuration memory, etc.

2.2 Status/Location Message (Message Type 0)

The message is used for reporting most of the basic unit events. It contains basic status data and location of the unit.

2.2.1 *Message Ingredients*

- Message Header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Communication Control Field – 2 bytes
  - Message Numerator – 1 byte
- Unit Hardware Version – 1 byte
- Unit Firmware Version – 1 byte
- Protocol Version and Unit Functionalities – 1 byte
- Unit Status – 1 byte
- Current GSM Operator – 2 bytes
- Transmission Reason Specific Data – 1 byte
- Transmission Reason – 1 byte
- Unit Mode of Operation – 1 byte
- Unit I/O status – 4 bytes
- Analog Input Values – 4 bytes
- Mileage Counter (Odometer) – 3 bytes
- Multi-Purpose Field – 6 bytes
- Last GPS Fix – 2 bytes
- Service and Status – 1 byte
- Mode 1/2 – 2 bytes
- Number of Satellites Used – 1 byte
- Longitude – 4 bytes
- Latitude – 4 bytes
- Altitude – 4 bytes
- Ground Speed – 4 bytes
- Speed Direction (True Course) – 2 bytes

- Time and Date – 7 bytes
- Error Detection Code – 1 byte

2.2.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type (0)
6	Unit ID
7	
8	
9	
10	Communication Control Field
11	
12	Message Numerator (Anti-Tango™)
13	Unit Hardware Version
14	Unit Firmware Version
15	Protocol Version and Unit Functionalities
16	Unit Status and Current GSM Operator (1st Nibble)
17	Current GSM Operator (2nd and 3rd Nibbles)
18	Transmission Reason Specific Data
19	Transmission Reason
20	Unit Mode of Operation
21	Unit I/O Status 1st byte
22	Unit I/O Status 2nd byte
23	Unit I/O Status 3rd byte
24	Unit I/O Status 4th byte
25	Current GSM Operator (4th and 5th Nibbles)

26	Analog Input 1 Value
27	Analog Input 2 Value
28	Analog Input 3 Value
29	Analog Input 4 Value
30	Mileage Counter (Odometer)
31	
32	
33	Multi-Purpose Field (Driver/Passenger/Group ID, PSP/Keyboard Specific Data, Accelerometer Status, SIM IMSI)
34	
35	
36	
37	
38	
39	Last GPS Fix
40	
41	Service and Status
42	Mode 1
43	Mode 2
44	Number of Satellites Used
45	Longitude
46	
47	
48	
49	Latitude
50	
51	
52	
53	Altitude
54	

55	
56	
57	Ground Speed
58	
59	
60	
61	Speed Direction (True Course)
62	
63	UTC Time – Seconds
64	UTC Time – Minutes
65	UTC Time – Hours
66	UTC Date – Day
67	UTC Date – Month
68	UTC Date – Year (-2000) (e.g. value of 7 = year 2007)
69	
70	Error Detection Code (8-bit additive checksum, excluding system code)

Multiple byte fields are sent Intel style (i.e. least significant bytes sent first).

2.2.3 *Detailed Per-Field Specifications*

2.2.3.1 System Code

System code is a 4-byte value, which identifies the Cellocator system. The field is sent as the ASCII values of the letters "M", "C", "G", "P" (for IP messages) or "M", "C", "G", "S" (for SMS messages), in that order.

2.2.3.2 Message Type

Message type identifies the kind of the message. It allows the receiver to differentiate between different messages types, according to the value sent in this field.

Status/Location messages contain a value of 0 (zero) in the message type field.

2.2.3.3 Unit ID

This field contains a value that is uniquely assigned for every Cellocator unit during the manufacturing process. All messages sent by the same unit contain the same value in the Unit ID field.

2.2.3.4 Communication Control Field

This is a bitmapped field, providing information about the message and the situation in which it was originated.

First byte (10th):

		Multi-Purpose Field (Bytes 33-38) assignment		Message Source	Garmin Connected	Garmin Enabled	Message Initiative
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Second byte (11th):

GSM Hibernation	Momentary /Max Speed	Business/ Private Mode	Firmware Sub-Version				
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message initiative

0 – Active transmissions (initiated by the unit, based on its logic and decisions)

1 – Passive responses (response to a command or a query message)

Message Source

0 – Direct message (not from memory)[1]

1 – Message from memory (the unit tries to resend the message from the memory, until ACK from the server is received)

Multi-Purpose Field (Bytes 33-38) Assignment

This 2 bits, along with bit 7 in byte 41 of this message (Service and Status), define the data provided in bytes 33-38 of this message according to the following table:

Byte 41	Byte 10		Data in Bytes 33-38
Bit 7	Bit 5	Bit 4	
0	0	0	Driver ID/Keyboard Code (for AR units)
1	0	0	IMEI
X	X	X	IMSI (in Wake Up event (TR 202))

[1] The only exception is the "Transmission Reason 32 - IP changed/Connection up" message, which always requires ACK from server, even if it was sent as a direct message and not through memory.

Note: The Communication Control Field is sent also in other (than 0) message types. In those message types the Multi-Purpose Field (Bytes 33-38) Assignment indication is a don't care.

Firmware Sub-Version

This field (5 bits) defines the firmware sub-version of the unit. The number of Cellocator firmware is built from two parts: [Firmware version][Firmware sub-version], where firmware version usually defines the unit family and the sub-version defines the list of supported features.

For example, 30a:

- Version – 30
- Sub-Version – a (1)

Firmware Sub-Version Value (decimal)	Firmware Sub-Version Identifier
0	No identifier
1	a
2	b
3	c
…	…
26	z

Business/Private Mode

It is possible to enable usage of Lock input as a Private/Business mode toggle. If enabled, every time the Lock input is triggered the unit switches to the opposite mode (Private → Business → Private). The default mode is Business. The Private mode is finished upon Lock input trigger, or when the active ID is erased from RAM after trip end.

0 – Bussiness

1 – Private

Momentary/Max Speed

0 – Momentary speed

1 – Max speed recorded from last event

Note: The Communication Control Field is sent also in other (than 0) message types. In those message types the Momentary/Max Speed indication is a “don't care”.

GSM Hibernation

0 – Unit is not in GSM hibernation

1 – Unit is in GSM hibernation (message sent during GSM peeking)

2.2.3.5 Message Numerator (Anti-Tango™)

The Message numerator field contains a value that is increased after every self-initiated generation of a message (in cases where an ACK from server was received).

When the unit is reset/powered-up, this value is set to zero. This provides a way to chronologically sort incoming messages from a certain unit, in case an anachronistic communication medium is used.

NOTE: The unit assigns different message numerator sequences for the logged events and for real-time events. In passive transmission (reply to command), the value in this field represents the number from the Command Numerator Field in an incoming command.

2.2.3.6 Unit Hardware Version

This field defines the unit HW (PCB) version and the ID of the modem embedded in it.

The legacy addressing scheme defined 5 bits for HW (PCB) ID and 3 bits for Modem Code. This limited the number of products to 32 products.

For new products (starting from CelloTrack Nano), an alternative backwards compatible approach will be used, in which each unit HW will be uniquely defined by a complete byte (8 bits).

The table for **legacy products**, which will be identified by the **Legacy HW ID** is detailed below:

New HW ID (8 Bits)	Legacy HW ID (5 Bits)	Product Name	Modem Code (3 Bits)	Modem Type
225	1	CR300	7	GE864-QUAD-V2
2	2	CFE	0	No Modem
170	10	CelloTrack 1 Output	5	Enfora 3
235	11	CR300B	7	GE864-QUAD-V2
172	12	CelloTrack	5	Enfora 3
78	14	Cello-IQ GNSS	2	GE910 QUAD V3
18	18	CelloTrack T (2G)	0	Telit GE910 QUAD (V2) (V3)
82	18	CelloTrack T (3G)	2	Telit HE910 NAD
114	18	CelloTrack T (3G)	3	Telit UE910 EUD
19	19	CelloTrackPower T (2G)	0	Telit GE910 QUAD (V2) (V3)
83	19	CelloTrackPower T (3G)	2	Telit HE910 NAD

New HW ID (8 Bits)	Legacy HW ID (5 Bits)	Product Name	Modem Code (3 Bits)	Modem Type
115	19	CelloTrackPower T (3G)	3	Telit UE910 EUD
20	20	Cello-CANiQ (NA)	0	UE910 NAR
52	20	Cello-CANiQ (EU)	1	UE910 EUR
84	20	Cello-CANiQ (2G)	2	GE910 QUAD V3
183	23	CelloTrack Power	5	Enfora 3
216	24	Cello-F (Telit)	6	Telit GE864, automotive
249	25	Cello-F Cinterion	7	Cinterion BGS3
221	29	CR200	6	Telit GE864, automotive
222	30	CR200B	6	Telit GE864, automotive
223	31	Cello-IQ	6	Telit GE864, automotive

The table for **new products**, which will be identified by the **New HW ID** is detailed below:

New HW ID (8 Bits)	Legacy HW ID (5 Bits)	Product Name	Modem Code (3 Bits)	Modem Type
38	6	Cello-D	1	UE910 NAR
70	6	Cello-D	2	UE910 EUR
136	8	CelloTrack Nano 10 GNSS	4	Cinterion BGS2-W
168	8	CelloTrack Nano 10 3G GNSS	5	Cinterion EHS6A
9	9	Cello-CANiQ CR (NA)	0	UE910 NAR
41	9	Cello-CANiQ CR (EU)	1	UE910 EUR
73	9	Cello-CANiQ CR (2G)	2	GE910 QUAD V3
105	9	Cello-CANiQ CR (2G) – Car Sharing	3	GE910 QUAD V3
169	9	Cello-CANiQ CR (3G) – Car Sharing	5	UE910 NAR
201	9	Cello-CANiQ CR (NA) – Aux	6	UE910 NAR

New HW ID (8 Bits)	Legacy HW ID (5 Bits)	Product Name	Modem Code (3 Bits)	Modem Type
233	9	Cello-CANiQ CR (EU) – Aux	7	UE910 EUR
43	11	CR300B 3G NA GNSS	1	UE910 NAD
75	11	CR300B 3G EU GNSS	2	UE910 EUD
107	11	CR300B 2G	3	GE910 QUAD V3
139	11	CR300B 2G SIRFV	4	GE910 QUAD V3
77	13	Cello-IQ CR GNSS	2	GE910 QUAD V3
15	15	CelloTrack 10Y	0	Cinterion ELS61-US
143	15	CelloTrack Solar	4	Cinterion ELS61-US
116	20	Cello-CANiQ (2G) – Car Sharing	3	GE910 QUAD V3
212	20	Cello-CANiQ CV	6	CE910 Dual V
244	20	Cello-CANiQ CS	7	CE910 Dual S
53	21	PointerCept Base Station	1	No Modem
88	24	Cello-CANiQ India (2G)	2	GE910 QUAD V3
26	26	CelloTrack Nano 20	0	Cinterion BGS2-W
122	26	CelloTrack Nano 20 3G Worldwide	3	Cinterion EHS6A
136	8	CelloTrack Nano 10 2G Worldwide	4	Cinterion BGS2-W
168	8	CelloTrack Nano 10 3G Worldwide	5	Cinterion EHS6A
218	26	CelloTrack Nano 20 LTE-Cat1 NA	6	Cinterion ELS61-USA R2
72	8	CelloTrack Nano 10 LTE-Cat1 NA	2	Cinterion ELS61-USA R2

2.2.3.7 Unit Firmware Version

This field defines the firmware version of the unit. The number of Cellocator firmware is built from two parts: [Firmware version][Firmware sub-version], where firmware version usually defines the unit family and the sub-version defines the list of supported features.

For example, 30a:

- Version – 30
- Sub-Version – a (1)

2.2.3.8 Protocol Version and Unit Functionalities

This is a bitmapped field, providing information about protocol version and other unit functionalities (AR, IQ).

				Protocol Version			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2.2.3.9 Unit Status and Current GSM Operator (1st Nibble)

This is a bitmapped field, providing information about unit statuses and current GSM operator.

Current GSM Operator (PLMN), 1st nibble				Source of Speed	Correct Time	Home/ Roam Network	GPS Comm.
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

GPS Comm.

0 – Communication with GPS is available

1 – Communication with GPS is not available

Home/Roam Network

0 – Home network

1 – Roam network

Correct Time

0 – Valid time stamp

1 – Invalid/estimated time stamp

Source of Speed

0 – GPS

1 – Pulse frequency input

Current GSM Operator

The current GSM Operator (PLMN) is represented as a 5 character hexadecimal number. After conversion into decimal it represents the MCC-MNC of a cellular operator (country code + network number). The 5 PLMN nibbles (nibble for each character) are provided in the following places:

Nibble 1	Nibble 2	Nibble 3	Nibble 4	Nibble 5
Byte 16	Byte 17		Byte 25	

(4MSbits)		

2.2.3.10 Current GSM Operator (2nd and 3rd Nibbles)

Current GSM Operator (PLMN), 2nd Nibble				Current GSM Operator (PLMN), 3rd Nibble			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2.2.3.11 Transmission Reason Specific Data

Additional information Related to the transmission reason (specified in byte 19)

<table>
<tr><th>Transmission Reason</th><th colspan="8">Transmission Reason Specific Data Description</th></tr>
<tr><td rowspan="3">8
Towing</td><td>0</td><td colspan="7">Location change detected during ignition off</td></tr>
<tr><td>1</td><td colspan="7">Towed mode start</td></tr>
<tr><td>2</td><td colspan="7">Towed mode stop</td></tr>
<tr><td rowspan="2">12
1-Wire Temperature Sensor Measurement Event</td><td colspan="4">0 - Low
1 - High</td><td colspan="4">Sensor ID (0-3)</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td rowspan="2">15
Crash detection</td><td colspan="2">Reserved</td><td>Light crash event</td><td>Heavy crash event</td><td colspan="4">Peak RMS value of the impact in 1g resolution minus 1g
(16g=0xF, 1g=0x0)</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>17
Hi-Res impact event</td><td colspan="8">RMS value sqrt($X^2+Y^2+Z^2$) of the impact in 16mg resolution (0.016g)</td></tr>
<tr><td>21
Coasting detection (speed and RPM)</td><td colspan="8">0 – Stop
1 – Start</td></tr>
</table>

<table>
<tr><th>Transmission Reason</th><th>Transmission Reason Specific Data Description</th></tr>
<tr><td>22
Violation of 1st Additional GP Frequency Threshold</td><td>0 – Falling
1 – Rising</td></tr>
<tr><td>23
Violation of 2nd Additional GP Frequency Threshold</td><td>0 – Falling
1 – Rising</td></tr>
<tr><td>34
Over speed start</td><td>0 – Plain
1 – Threshold changed by input</td></tr>
<tr><td>42
Over speed end</td><td>0 – Plain
1 – Threshold changed by input</td></tr>
<tr><td>46
Driver authentication update</td><td>
<table>
<tr><td>Group ID status
0 – Not Authenticated
1 – Authenticated</td><td>Unused</td><td>Unused
Driver ID Card Introduced/ Removed
0 – Introduced
1 - Removed</td><td>User Type
0 – Driver ID
1 – Passenger ID</td></tr>
<tr><td>Bit 7</td><td>Bits 2-6</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
NOTE: If "Enable Pre-defined driver ID list" parameter (address 123, bit 2) is enabled and the ID is not authenticated, The 6 bytes Dallas field must be ignored.</td></tr>
<tr><td>47
Driving without authentication</td><td>0 - Legacy logics</td></tr>
</table>

<table>
<tr><th>Transmission Reason</th><th>Transmission Reason Specific Data Description</th></tr>
<tr><td>Door
48 - Close
64 - Open</td><td>0 – Normal
1 – Robbery Event
2 – Car Sharing 2: End Of Reservation</td></tr>
<tr><td>Shock/Unlock2

49 - Inactive
65 - Active</td><td>0 – Normal
1 – Car Sharing 2: Modem Off Ended
2 – Car Sharing 2: Modem Off Started
3 – Car Sharing 2: Business Mode started
4 – Car Sharing 2: Private Mode started</td></tr>
<tr><td>53
Driving stop</td><td>0 – Accelerometer based</td></tr>
<tr><td>69
Driving start</td><td>0 – Accelerometer based
1 – GPS based (CelloTrack family only)</td></tr>
<tr><td>158
Tamper active</td><td>1 – Reserved
2 – Nano and PointerCept(CR300): Tilt tamper</td></tr>
<tr><td>191
Geo hotspot violation</td><td>
<table>
<tr><td>Direction
0 – exit from hot spot
1 – entry to hot spot</td><td colspan="7">The index of the geo-fence</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Transmission Reason</th><th>Transmission Reason Specific Data Description</th></tr>
<tr><td>192
Frequency measurement threshold violation</td><td>
<table>
<tr><td rowspan="2">Violating input number

0 – Door
1 – Shock</td><td rowspan="2">Violation status

0 – Violation start
1 – Violation End</td><td rowspan="2">Violation type

0 – Threshold
1 – Range</td><td>Violation direction

In case of Threshold
0 – Low threshold
1 – High threshold</td><td colspan="4" rowspan="2">Reserved</td></tr>
<tr><td>In case of range
0 – Keep In
1 – Keep Out</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
<tr><td>194
Analog measurement threshold violation</td><td>
<table>
<tr><td>Violating input number

0 – Door
1 – Shock</td><td>Violation status

0 – Violation start
1 – Violation End</td><td>Violation type

0 – Threshold
1 – Range</td><td>Violation direction

0 – Low threshold
1 – High threshold</td><td colspan="4">Reserved</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
<tr><td>199
Trailer connection status</td><td>Trailer Connection Status
0 – Trailer disconnected
1 – Trailer connected</td></tr>
</table>

<table>
<tr><th>Transmission Reason</th><th>Transmission Reason Specific Data Description</th></tr>
<tr><td>200
AHR (Auto Hardware Reset)</td><td>
<table>
<tr><td colspan="4">AHR reason
0 – Modem non responsiveness
1 – Registration problem
2 – GPS AHR</td><td colspan="4">Number of performed AHR attempts</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
<tr><td>205
Garmin connection status</td><td>0 – Garmin disconnected
1 – Garmin connected</td></tr>
<tr><td>206
Jamming detection</td><td>
<table>
<tr><td colspan="5">Not used</td><td colspan="2">GSM jamming ignition state
0 – Legacy (Not associated with Ignition state)
Advanced Jamming Mode:
1 – Ignition Off
2 – Ignition On</td><td>GSM jamming state
0 – GSM jamming detection start
1 – GSM jamming detection end</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
<tr><td>207
Radio off mode</td><td>
<table>
<tr><td colspan="2">Spare</td><td></td><td></td><td>Airplane Mode</td><td>Early Radio Off Event</td><td>GPS Status
0 – Off
1 – On</td><td>Modem Status
0 – Off
1 – On</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
<table>
<tr><th>(Bit 5)</th><th>(bit 4)</th><th>Airplane Mode
(Bit 3)</th><th>Early Radio Off Event
(Bit 2)</th><th>GPS Status
(Bit 1)</th><th>Modem Status
(Bit 0)</th><th>Description</th></tr>
</table>
</td></tr>
</table>

Transmission Reason	Transmission Reason Specific Data Description						
	0		0	0	0	0	Detection of internal backup battery voltage discharging to 3.25V or below for longer than 1 second (100 samples). The unit will enter shipment mode only after generating this event.
	0	0	0	0	0	1	N/A
	0	0	0	0	1	0	Detection of internal backup battery voltage lower than 3.46V (on any temperature) for longer than 1 second (100 samples) upon sole work from internal backup battery. The unit will switch off the radio 2 seconds after event generation. Once switched off, the modem will be switched back on only upon main power reconnection.
	0	0	0	0	1	1	N/A
	0	0	0	1	0	0	N/A
	0	0	0	1	0	1	N/A
	0	0	0	1	1	0	N/A

<table>
<tr><th>Transmission Reason</th><th>Transmission Reason Specific Data Description</th></tr>
<tr><td></td><td><table><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>N/A</td></tr></table></td></tr>
<tr><td>212
Geo-fence over speed start</td><td rowspan="2">Index of the geo-fence</td></tr>
<tr><td>213
Geo-fence over speed end</td></tr>
<tr><td>222
PointerCept beacon</td><td>0 – Reserved
1 – PointerCept operational beacon transmission start
2 – PointerCept operational beacon transmission stop
3 – PointerCept OTA command initiated beacon transmission start
4 – PointerCept OTA command initiated beacon transmission stop
5 – PointerCept periodic beacon transmission start
6 – PointerCept periodic beacon transmission stop</td></tr>
<tr><td>252
Com location glancing / Offline tracking</td><td>3 – logged events upload due to timer expiration
4 – logged events upload due to full memory
5 – logged events upload due to events amount
6 – logged events upload due to end of trip
7 – logged events upload due to input activation

9 – Offline tracking local timer glancing
10 – logged events upload due to Modem FOTA process</td></tr>
<tr><td>253
Violation of</td><td>Index of the geo-fence</td></tr>
</table>

Transmission Reason	Transmission Reason Specific Data Description
keep in fence	
254 Violation of keep out fence	
255 Violation of waypoint	

2.2.3.12 Transmission Reason

This field contains the reason for the message transmission. Note that this value is valid only for self-initiated active transmissions, i.e. transmissions that the unit generated because of its logics, in contrast to reply transmissions. Reply transmissions contain the last transmission reason that was used.

Transmission Reason Value	Transmission Reason Description
4	Emergency (Distress) mode by command
6	Engine activated (security event)
8	Towing
11	Communication idle
12	1-Wire Temperature Sensor Measurement Event
15	Crash detection
17	Hi-Res impact event
21	Coasting detection (speed and RPM)
22	Violation of 1st additional GP frequency threshold
23	Violation of 2nd additional GP frequency threshold
25	Speed detected during ignition off
31	Reply to command
32[2]	IP changed/connection up
33	GPS navigation start
34	Over speed start

[2] Always requires acknowledge from server, even if it was sent as a direct message and not through memory.

Transmission Reason Value	Transmission Reason Description
35	Idle speed start
36	Distance event
37	Engine start; ignition input – active (high)
38	GPS factory reset (automatic only)
41	GPS navigation end
42	Over speed end
43	Idle speed end
44	Timed event[3]
45	Engine stop; ignition input – inactive (low)
46	Driver authentication update
47	Driving without authentication
48	Door close
49	Shock/Unlock2 inactive
51	Volume sensor inactive event
53	Driving stop
54	Distress button inactive
63	Ignition input inactive
64	Door open
65	Shock/Unlock2 active
67	Volume sensor active
69	Driving start
70	Distress button active
79	Ignition input active or CFE input 6 active
80	Main power disconnected
81	Main power low level
82	Backup battery disconnected

[3] In Cello-CANiQ, this event is used also for the 1 second GPS data reporting.

Transmission Reason Value	Transmission Reason Description
83	Backup battery low level
84	Halt (movement end)
85	Go (movement start)
87	Main power connected (unconditionally logged upon an initial power up)
88	Main power high level
89	Backup battery connected
90	Backup battery high level
92	Satellite communication
158	Tamper Active
190	No Modem zone entry
191	Geo hotspot violation
192	Frequency measurement threshold violation
194	Analog measurement threshold violation
199	Trailer connection status
200	AHR (Auto Hardware Reset)
201	PSP – External Alarm is Triggered
202	Wake Up event
203	Pre-hibernation event
204	Vector (course) change (curve smoothing event)
205	Garmin connection status
206	Jamming detection
207	Radio off mode
208	Header error (self re-flash processing)
212	Geo-fence over speed start
213	Geo-fence over speed end
222	PointerCept beacon start/stop
223	PointerCept CPIN error event

Transmission Reason Value	Transmission Reason Description
224	OTA command initiated PointerCept beacon (will be transmitted via RF only)
225	PointerCept periodic beacon transmission (will be transmitted via RF only)
247	Finish mode
253	Violation of keep in fence
254	Violation of keep out fence
255	Violation of waypoint

2.2.3.13 Unit Mode of Operation

The functioning of the unit can be generalized as a finite state machine model, with a few "stages" of operation. The "current stage" is referred to as "unit mode", or "mode of operation", as following:

Unit Mode Value	Unit Mode Description
0x00	Standby Engine On
0x01	Standby Engine Off
0x10	Towed mode (same as Standby Engine On, but with ignition off)

2.2.3.14 Unit I/O Status

The unit is provided with many I/Os (inputs/outputs). Each I/O may be "high" or "low" at a given moment. The I/O status field is a bitmapped representation of the I/Os physical levels. Note that the I/Os that have been configured to be inverted will affect the application but will not be shown in this field, as it only represent the raw physical signals read from the HW.

1st Byte of I/O Status

CR300/ CR300B			Driving Status (physical ignition or accelerometer based)				Shock	Door
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Note: Driving Status (bit 5) provides indication if the unit is in logical Ignition On/Off, according to the configuration of the detection source (physical ignition or accelerometer). It will indicate "1" when logical Ignition On is detected, and "0" when logical Ignition Off is detected.

2nd Byte of I/O Status

CR300/ CR300B	Ignition port status	Accelerometer status						
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Notes: Accelerometer Status (bit 6) provides indication if the accelerometer has detected Ignition On/Off, **regardless** of the configuration of the detection source (physical ignition or accelerometer). It will indicate "1" when accelerometer Ignition On is detected, and "0" when accelerometer Ignition Off is detected. Ignition Port Status (bit 7) provides indication if the physical ignition input is high/low, **regardless** of the configuration of the detection source (physical ignition or accelerometer). It will indicate "1" when the ignition input is high, and "0" when the ignition input is low.

3rd Byte of I/O Status

CR300/ CR300B					GPS Power	Grad. Stop		-
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

4th Byte of I/O Status

CR300/ CR300B	Charger status		Standard Immobilizer		Blinkers/ Unlock			LED out/ Lock
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2.2.3.15 Current GSM Operator (4th and 5th Nibbles)

Current GSM Operator (PLMN), 4th Nibble				Current GSM Operator (PLMN), 5th Nibble			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2.2.3.16 Analog Inputs

The unit may handle up to 4 analog inputs. These inputs are multiplexed and sent as 8 bit samples each.

The allocation of measurements to the bytes of the message is configurable (PL addresses 1620-1623).

For Cello/CR300 unit:

Field name	Default value	Byte number in the message
1st analog measurement	9 (Vin)	26

2^{nd} analog measurement	6 (Vbat)	27
3^{rd} analog measurement	7 (Bat. NTC)	28
4^{th} analog measurement	2 (Shock)	29

Available inputs for mapping:

Measurement source number	Measurement source name	Coefficient	Comment
0	No source		
1	Door[4]	0.009801587 [2.5V] 0.117619048 [30V]	Can report either analog or frequency measurement as per corresponding input type
2	Shock	0.009801587 [2.5V] 0.117619048 [30V]	
6	V bat	0.01647058823	Battery voltage
7	Bat. NTC	Temperature conversion formula: T=0.4314x-40; 0≤x≤255 (-40ºC≤T≤70ºC)	Note that the accuracy of the measurement is ±3°C
8	V main	0.0176470588235	Regulated voltage
9	V in	0.1176470588235	Input voltage
16	1-Wire temperature sensor 1	Signed 8	
20	RSSI		

2.2.3.17 Mileage Counter (Odometer)

The unit is provided with a distance accumulator feature. The unit counts distance "base units" programmed in the PL.

By synchronizing the accumulator value with the vehicle odometer reading and setting the distance base units to one kilometer/mile, this counter provides the ability to remotely read the vehicle odometer. The programming and synchronizing is only needed once – during the installation.

The mileage counter field contains the current 24-bit value of this accumulator.

[4] The analog inputs measurement resolution is variable (either in 9.8mA or 117.6mA resolution), and controlled by programmable parameter.

2.2.3.18 Multi-Purpose Field (Bytes 33-38)

This field may carry different information as per bits 4, 5 in Communication Control Field (byte 10) and bit 7 in Service and Status (byte 41):

Byte 41	Byte 10		Data in Bytes 33-38
Bit 7	Bit 5	Bit 4	
0	0	0	Driver ID/ Keyboard Code (for AR units)
1	0	0	IMEI
X	X	X	IMSI (in Wake Up event (TR 202))

Driver ID/Passenger ID/Group ID Code Update

The unit can provide 6 bytes of last received Dallas button in every message if that feature is enabled in PL (Mask of Authentication Events).

If no Dallas code is received since the initiation of the last Start Event, this field will be 0.

The code can carry Driver ID or Passenger ID and Group ID, depends on the type of the attached button and the configuration.

Group ID

The Group ID is an additional driver authentication method, used when there are too many drivers to be programmed into unit memory.

The length of Group ID varies from 1 to 9 bytes length but shorter than 10 digits. The unit supports multiple groups, while all Group IDs are from the same length.

NOTE: Group ID number will never begin from zero.

The first number in Dallas codes array, shorter than 10 digits is considered as group ID and its length is considered length of group ID. Any additional number, shorter than 10 digits but with length different from the first Group ID length, is considered a Driver ID.

Example: Dallas code 1234567890, when group ID is 4 digits:

Driver/Passenger ID 567890			Group ID 1234		
90	78	56	34	12	00
Byte 33	Byte 34	Byte 35	Byte 36	Byte 37	Byte 38

IMEI

Will be sent on bytes 33-38 with its 2 MS-Bits sent in bits 5, 6 in byte 41 of this message (Service and Status).

IMEI is defined as 15 decimal digits. Converting the maximal IMEI number 999999999999999 to hexadecimal we get: 38D7EA4C67FFF. The maximal number will occupy 50 bits which will be sent as follows:

0x03	0xFF	0x7F	0xC6	0xA4	0x7E	0x8D
Byte 41, bits 5, 6	Byte 33	Byte 34	Byte 35	Byte 36	Byte 37	Byte 38

Note: for CDMA devices, the IMEI is replaced with MEID, which is 18 decimal digits long. Thus, MEID will not be transmitted in these bytes (only in Type 9, sub data 0x12).

IMSI

In case of a Wake Up event (TR 202), the unit reports the 12 first characters of the SIM IMSI converted to hex (Little Endian).

The IMSI number consists of up to 15 numerical characters (0-9). An IMSI consists of a three digit mobile country code (MCC, which is not reported by Cellocator Protocol) and a variable length national mobile station identity (NMSI).

The NMSI consists of two variable length parts: the mobile network code (MNC) and the mobile station identification number (MSIN). A Class 0 IMSI is 15 digits in length. A Class 1 IMSI is less than 15 digits in length.

Example: 425020315229000 (Cellcom IL)

MCC	425	Israel
MNC	02	Cellcom IL
MSIN	0315229000	

The Hex value received in bytes 33-38:

Value (hex)	00	5A	16	0F	03	02
Location	Byte 33	Byte 34	Byte 35	Byte 36	Byte 37	Byte 38

Conversion table:

In wireless protocol (big-endian)	00	5A	16	0F	03	02
HEX values (little-endian)	02	03	0F	16	5A	00
DEC values	02	03	15	22	90	00
NMSI (MNS + MSIN)	020315229000					

2.2.3.19 Last GPS Fix

This field provides a timestamp when which the GPS was last in navigation mode.

Day of Month					Hours					Minutes					
Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 40								Byte 39							

NOTE: The easiest way to define if the GPS data in the message is valid and updated, or historical, is to compare between the time of the timestamps and UTC time (see below).

2.2.3.20 Service and Status

MSB of Multi-Purpose field (bytes 33-38) assignment (with bits 4, 5 of byte 10)	IMEI Bit 49	IMEI Bit 48					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2.2.3.21 MODE 1 and Mode 2

These fields are generated by the GPS and transparently monitored in the outgoing message from the unit. The fields define the validity of GPS data in the message.

The unit considers the valid fix according to the "Enable Tight GPS PMODE Filter" parameter (address 509, bit 7):

- If "Enable Tight GPS PMODE Filter" is enabled, the unit considers the GPS data as valid only if Mode 1 = 3 or 4 AND Mode 2 = 2
- If "Enable Tight GPS PMODE Filter" is disabled, the unit considers the GPS data as valid only if Mode 1 =2, 3, 4, 5 and 6

2.2.3.22 Number of Satellites Used

Number of satellite measurements used for current position fix. Possible values are 0 to 20 (GNSS modules).

2.2.3.23 Longitude, Latitude

Longitude and latitude coordinates of current position fix. Both coordinates are sent as 32-bit signed integers, representing the coordinates in 10^{-8} radian resolution. Possible values are $-\Pi$ to $+\Pi$ for longitude, or $-\Pi/2$ to $+\Pi/2$ for latitude. The coordinates refer to WGS-84 map datum and ellipsoid.

2.2.3.24 Altitude

Altitude of current position fix. Represented as a 32-bit signed integer, in 10^{-2} meter resolution (altitude is represented in centimeters).

2.2.3.25 Ground Speed

Current speed (absolute value of the vector). Represented as a 32-bit unsigned integer, in 10^{-2} meter/sec resolution (speed is represented in centimeters/sec).

2.2.3.26 Heading/Speed Direction (True Course)

Direction (angle) of the speed vector. Represented as 16-bit unsigned integer, in 10^{-3} radian resolution. Possible values are 0 to 2Π.

2.2.3.27 System Time

Universal coordinated time of the position fix, represented in seconds (0-59), minutes (0-59) and hours (0-23).

Note that the system time and date fields are monitoring system time, based on the internal timer of the unit. The internal timer synchronizes with GPS time when the GPS fix is considered as valid (or always as per configuration flag).

2.2.3.28 System Date

Universal coordinated date of the position fix, represented in days (1-31), months (1-12) and years (1980-2079).

Note that the system time and date fields are monitoring system time, based on the internal timer of the unit. The internal timer synchronizes with GPS time when the GPS fix is considered as valid (or always as per configuration flag).

2.2.3.29 Error Detection Code

The error detection code (checksum) is a last byte of sum of all bytes in a message, excluding the 4 bytes of System Code and the Error Detection Code itself.

Example:
The message:
*4D434750*0006000000081A020212040000002100623000000006B00E1000000000000000
00000E5A100040206614EA303181A57034E120000000000000000001525071403D607*CS*

Calculation of the CS=>

00+06+00+00+00+08+1A+02+02+12+04+00+00+00+21+00+62+30+00+00+00+6B
+00+E1+00+00+00+00+00+00+00+00+00+00+E5+A1+00+04+02+06+61+4E+A3+0
3+18+1A+57+03+4E+12+00+00+00+00+00+00+00+00+15+25+07+14+03+D6+07=
0x749

=>CS=0x49

2.2.4 *Distress (Emergency) Queue Description*

There is a dedicated queue in size of 5 for distress (emergency) messages.

In this queue, if new emergency events with the same TR which exist in the queue occur, the older event is replaced by the new one.

2.3 Programming Data (Message Type 3)

This message is sent as a reply to programming commands, or by request. It contains the new contents of the programmed block.

NOTE: For configuration spaces larger than 4K (Typically in Cello-IQ and Cello-CANiQ units) it is mandatory to use Type 11 programming command (modules 10, 11).

2.3.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Communication Control Field – 2 bytes
  - Message Numerator – 1 byte
- Spare – 1 byte
- Block Code – 1 byte
- Block Data – 16 bytes
- Error Detection Code – 1 byte

2.3.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type (3)
6	Unit ID
7	
8	
9	
10	Communication Control Field
11	

12	Message Numerator (Anti-Tango™)
13	Spare
14	Block Code
15-30	Block Data
31	Error Detection Code (8-bit additive checksum, excluding system code)

2.3.3 *Detailed Per-Field Specifications*

2.3.3.1 System Code

Refer to Section 2.2.3.1

2.3.3.2 Message Type

Programming Data messages contain a value of 3 (three) in the message type field.

2.3.3.3 Unit ID

Refer to Section 2.2.3.3

2.3.3.4 Communication Control Field

Refer to Section 2.2.3.4

2.3.3.5 Message Numerator (Anti-Tango™)

Refer to Section 2.2.3.5

2.3.3.6 Block Code

OTA (over the air) parameter programming is done in blocks. The entire parameter memory is partitioned to 16-bytes long blocks. Each of those blocks is identified with a block code. The block code field contains the code of the block whose data is sent in this message (in the block data field).

2.3.3.7 Block Data

Contains the actual data programmed in the specified block of the parameter memory.

2.4 Logged Fragment of Forwarded Data from Serial Port to Wireless Channel (Message Type 7)

The unit can forward data from its serial port to the OTA channel in a logged or in real time manner.

If the unit is configured to work with logged forwarding ("Enable Data forwarding through log" parameter (address 285, bit 7) is enabled), message type 7 will be used. Message type 7 contains fragments (up to 54 bytes each) of payload forwarded from the unit serial port.

If the unit is configured to work with real time forwarding ("Enable Data forwarding through log" parameter (address 285, bit 7) is disabled), message type 8 will be used. Message Type 8 contains a complete payload (up to 512 bytes) forwarded from the unit serial port.

The forwarded payload may be escorted by fleet management data (as per unit configuration).

Like other message types which are utilizing log memory (e.g. 0 and 9), message type 7:

- Continues the Message Numerator used by other logged messages.
- Requires acknowledge from the server (Message type 4) in order to erase the specific message from the log.
- Utilizes the same retransmission algorithms as other logged message types.

2.4.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Communication Control Field – 2 bytes
  - Message Numerator – 1 byte
- Serial Port Source – 1 byte
- Forwarded Message Code – 1 byte
- Fragment Control Byte – 1 byte
- Container Fragment – 54 bytes
- Error Detection Code – 1 byte

2.4.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>2</td><td>System Code, byte 2 – ASCII “C”</td></tr>
<tr><td>3</td><td>System Code, byte 3 – ASCII “G”</td></tr>
<tr><td>4</td><td>System Code, byte 4 – ASCII “P”</td></tr>
<tr><td>5</td><td>Message Type (7)</td></tr>
<tr><td>6</td><td rowspan="4">Unit ID</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td><td rowspan="2">Communication Control Field</td></tr>
<tr><td>11</td></tr>
<tr><td>12</td><td>Message Numerator (Anti-Tango™)</td></tr>
<tr><td>13</td><td>Serial Port Source
<table>
<tr><td colspan="3">Source of Payload
0 – N/A
1 – N/A
2 – COM2 (BT)
3 – COM3
4 – COM4
5 – COM5
6 – CFE Micro
7 – N/A</td><td>CFE Connected
0 – Not connected
1 – Connected</td><td colspan="4">Static nibble containing value 0x07</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>14</td><td>Forwarded Message Code
Sequential 7 bits ID of the container + container indication bit (MSB)
Assigned for each container
<table>
<tr><td>0 – Simple payload
1 – Container</td><td colspan="7">In case of container: sequential 7 bits ID of the container
In case of simple payload: sequential 7 bits ID of the forwarded packet</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 6</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>15</td><td>Fragment Control Byte
<table>
<tr><td>First Fragment
0 – Not first
1 - First</td><td>Last Fragment
0 – Not last
1 - Last</td><td colspan="6">Fragment No (starting from 1)</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>16-69</td><td>Container Fragment
(first fragment begins with two bytes of length of container, last one is zero padded)</td></tr>
<tr><td>70</td><td>Error Detection Code (8-bit additive checksum, excluding system code)</td></tr>
</table>

2.4.3 *Detailed Per-Field Specifications*

2.4.3.1 System Code

Refer to Section 2.2.3.1

2.4.3.2 Message Type

Logged Fragment of Forwarded Data from Serial Port to Wireless Channel messages contain a value of 7 (seven) in the message type field.

2.4.3.3 Unit ID

Refer to Section 2.2.3.3

2.4.3.4 Communication Control Field

Refer to Section 2.2.3.4

2.4.3.5 Message Numerator (Anti-Tango™)

Refer to Section 2.2.3.5

2.4.3.6 Serial Port Source

This field provides information about the source of data connected to the unit serial port.

2.4.3.7 Forwarded Message Code

This field provides information about the container in the message.

If the unit is configured to work with container ("Forward Data as Container" parameter (address 285, bit 6) is enabled), the payload will be in a form of a container: forwarded payload from serial port is escorted by 48 bytes of FM (fleet management) data, and 2 bytes of total length of payload + FM data.

If the unit is configured to work with simple payload ("Forward Data as Container" parameter (address 285, bit 6) is disabled), the payload will be in a form of a simple payload: forwarded payload from serial port only.

In addition, this byte includes a container/simple payload sequential ID.

2.4.3.8 Fragment Control Byte

This field provides information about the current payload fragment.

2.4.3.9 Container Fragment

The container is a data structure, created by the unit in its RAM buffer upon reception of the data for forwarding from the unit serial port (if enabled in "Forward Data as Container" parameter (address 285, bit 6)).

The forwarded payload from serial port is escorted by 48 bytes of FM (fleet management) data, and 2 bytes of total length of payload + FM data.

Every container is assigned by 7 bits numerator (increased every data packet received from the serial port), used in fragmentation process and reported with the container.

The container data structure is as following:

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>1</td><td rowspan="2">Payload length (X)</td></tr>
<tr><td>2</td></tr>
<tr><td>3</td><td rowspan="3">Forwarded Payload from serial port, X bytes (up to 512 bytes)</td></tr>
<tr><td></td></tr>
<tr><td>3+X</td></tr>
<tr><td>4+X</td><td>Unit Status + Current GSM Operator (1st nibble) (same as byte 16 of type 0)</td></tr>
<tr><td>5+X</td><td>Current GSM Operator (2nd and 3rd nibbles) (same as byte 17 of type 0)</td></tr>
<tr><td>6+X</td><td>Current GSM Operator (4th and 5th nibbles) (same as byte 25 of type 0)</td></tr>
<tr><td>7+X</td><td>Unit Mode of Operation (same as byte 20 of type 0)</td></tr>
<tr><td>8+X</td><td>Unit I/O Status 1st byte (same as byte 21 of type 0)</td></tr>
<tr><td>9+X</td><td>Unit I/O Status 2nd byte (same as byte 22 of type 0)</td></tr>
<tr><td>10+X</td><td>Unit I/O Status 3rd byte (same as byte 23 of type 0)</td></tr>
<tr><td>11+X</td><td>Unit I/O Status 4th byte (same as byte 24 of type 0)</td></tr>
<tr><td>12+X</td><td>Analog Input 1 value (same as byte 26 of type 0)</td></tr>
<tr><td>13+X</td><td>Analog Input 2 Value (same as byte 27 of type 0)</td></tr>
</table>

14+X	Analog Input 3 Value (same as byte 28 of type 0)
15+X	Analog Input 4 Value (same as byte 29 of type 0)
16+X	Mileage Counter (Odometer) (same as bytes 30-32 of type 0)
17+X	
18+X	
19+X	Multi-Purpose Field (Driver/Passenger/Group ID, PSP/Keyboard Specific Data, Accelerometer Status, SIM IMSI) (same as bytes 33-38 of type 0)
20+X	
21+X	
22+X	
23+X	
24+X	
25+X	Last GPS Fix (same as bytes 39-40 of type 0)
26+X	
27+X	Location Status (flags) (same as sub type 4 of type 9)
28+X	Mode 1
29+X	Mode 2
30+X	Number of Satellites Used
31+X	Longitude
32+X	
33+X	
34+X	
35+X	Latitude
36+X	
37+X	
38+X	
39+X	Altitude
40+X	
41+X	
42+X	Ground speed

43+X	
44+X	Speed direction (true course)
45+X	
46+X	UTC time – Seconds
47+X	UTC time – Minutes
48+X	UTC time – Hours
49+X	UTC date – Day
50+X	UTC date - Month
51+X	UTC date - Year (-2000) (e.g. value of 7 = year 2007)

2.5 Real Time Forwarded Data from Serial Port to Wireless Channel (Message Type 8)

The unit can forward data from its serial port to the OTA channel in a logged or in real time manner.

If the unit is configured to work with logged forwarding ("Enable Data forwarding through log" parameter (address 285, bit 7) is enabled), message type 7 will be used. Message type 7 contains fragments (up to 54 bytes each) of payload forwarded from the unit serial port.

If the unit is configured to work with real time forwarding ("Enable Data forwarding through log" parameter (address 285, bit 7) is disabled), message type 8 will be used. Message Type 8 contains a complete payload (up to 512 bytes) forwarded from the unit serial port.

The forwarded payload may be escorted by fleet management data (as per unit configuration).

2.5.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Message Numerator – 1 byte
- Spare – 2 bytes
- Serial Port Source – 1 byte
- Spare – 1 byte
- Forwarded Message Code – 1 byte
- Fragment Control Byte – 1 byte
- Payload Length – 2 bytes
- Payload – variable length
- Error Detection Code – 1 byte

2.5.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>4</td><td>System Code, byte 4 – ASCII “P”</td></tr>
<tr><td>5</td><td>Message Type (8)</td></tr>
<tr><td>6</td><td rowspan="4">Unit ID</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td><td>Message Numerator (Anti-Tango™)</td></tr>
<tr><td>11</td><td rowspan="2">Spare</td></tr>
<tr><td>12</td></tr>
<tr><td>13</td><td>Serial Port Source
<table>
<tr><td colspan="3">Source of Payload
0 – N/A
1 – N/A
2 – COM2 (BT)
3 – COM3
4 – COM4
5 – COM5
6 – CFE Micro
7 – N/A</td><td>CFE Connected
0 – Not connected
1 – Connected</td><td colspan="4">Static nibble containing value 0x07</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>14</td><td>Spare</td></tr>
<tr><td>15</td><td>Forwarded Message Code
Sequential 7 bits ID of the container + container indication bit (MSB)
Assigned for each container
<table>
<tr><td>0 – Simple payload
1 – Container</td><td colspan="7">In case of container: sequential 7 bits ID of the container
In case of simple payload: sequential 7 bits ID of the forwarded packet</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 6</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>16</td><td>Fragment Control Byte
<table>
<tr><td>First Fragment
0 – Not first
1 - First</td><td>Last Fragment
0 – Not last
1 - Last</td><td colspan="6">Fragment No (starting from 1)</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>17</td><td rowspan="2">Payload Length</td></tr>
<tr><td>18</td></tr>
<tr><td>…</td><td>Payload</td></tr>
<tr><td>…</td><td>Error Detection Code (8-bit additive checksum, excluding system code)</td></tr>
</table>

2.5.3 *Detailed Per-Field Specifications*

2.5.3.1 System Code

Refer to Section 2.2.3.1

2.5.3.2 Message Type

Logged Fragment of Forwarded Data from Serial Port to Wireless Channel messages contain a value of 8 (eight) in the message type field.

2.5.3.3 Unit ID

Refer to Section 2.2.3.3

2.5.3.4 Message Numerator (Anti-Tango™)

Refer to Section 2.2.3.5

2.5.3.5 Serial Port Source

This field provides information about the source of data connected to the unit serial port.

2.5.3.6 Forwarded Message Code

This field provides information about the container in the message.

If the unit is configured to work with container ("Forward Data as Container" parameter (address 285, bit 6) is enabled), the payload will be in a form of a container: forwarded payload from serial port is escorted by 48 bytes of FM (fleet management) data, and 2 bytes of total length of payload + FM data.

If the unit is configured to work with simple payload ("Forward Data as Container" parameter (address 285, bit 6) is disabled), the payload will be in a form of a simple payload: forwarded payload from serial port only.

In addition, this byte includes a container/simple payload sequential ID.

2.5.3.7 Fragment Control Byte

This field provides information about the current payload fragment.

The current implementation of message type 8 allows to send the payload in a single message (i.e. without fragmentation). However, for backward compatibility reasons, there is an option to fragment the payload.

If the unit is configured to work with the extended implementation ("Backward compatible OTA msg type 8" parameter (address 1349, bit 2) = extended), the payload will be sent in single type 8 message (up to 512 bytes payload). In this case, the fragment control byte will be set to 0xC0.

If the unit is configured to work with the backward compatible implementation ("Backward compatible OTA msg type 8" parameter (address 1349, bit 2) = backward compatible), the payload will be sent in fragmented type 8 messages (up to 235 bytes payload, up to 82 bytes per fragment). In this case, the fragment control byte will be used normally.

2.6 Modular Message (Message Type 9)

The modular data packet is designed to provide different data types in the same message.

2.6.1 *Message Ingredients*

- Message Header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Communication Control Field – 2 bytes
  - Message Numerator – 1 byte
- Packet Control Field – 1 byte
- Message Length – 1 byte
- First Sub-Data Type – 1 byte
- First Sub-Data Length – 1 byte
- First Sub-Data variable length, depends on Data Type
- ……
- Nth Sub-Data Type – 1 byte
- Nth Sub-Data Length – 1 byte
- Nth Sub-Data– variable length, depends on Data Type N
- Error Detection Code – 1 byte

2.6.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”
4	System Code, byte 4 – ASCII “P”
5	Message Type (9)
6	Unit ID
7	
8	

9	
10	Communication Control field
11	
12	Message Numerator
13	Packet Control Field
14	Length (of the modules section - not including the checksum)
15	First Sub-data Type
16	First Sub-data Length
17	First Sub-data The Data
…	…
	Nth Sub-data Type
	Nth Sub-data Length
	Nth Sub-data The Data
Last Byte	Error Detection Code (8-bit additive checksum, excluding system code)

2.6.3 *Detailed Per-Field Specifications*

2.6.3.1 System Code

Refer to Section 2.2.3.1

2.6.3.2 Message Type

Modular messages contain a value of 9 (nine) in the message type field.

2.6.3.3 Unit ID

Refer to Section 2.2.3.3

2.6.3.4 Communication Control Field

Refer to Section 2.2.3.4

2.6.3.5 Message Numerator (Anti-Tango™)

Refer to Section 2.2.3.5

2.6.3.6 Packet Control Field

Direction	Out of space indication	Unused					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Direction

0 – Data from the unit

1 – Request (unit-bound)

Out of Space Indication

0 – All the requested data present in the message

1 – Some Sub-Data was not returned due to data size

2.6.3.7 Length

That field includes the number of data bytes with their types and lengths. It includes the number of bytes from byte 15 to the byte of the checksum, which is not included.

2.6.4 *Outbound Sub-Data Types Table*

Code (Hex)	Function
0x01	Firmware Platform Manifest
0x04	Time and Location Stamp
0x07	Usage Counter
0x08	Authentication Table Update
0x09	Neighbor List of the Serving GSM Cell
0x0A	Maintenance Server Platform Manifest
0x0C	3G Cell ID Data
0x0D	Compressed vector change report
0x12	Modular Platform Manifest
0x14	Pulse Counter Measurement
0x19	One-Wire Temperature Measurements

2.6.5 *Firmware Platform Manifest*

This sub-data is generated as a reply to Firmware Platform Manifest Request (0x01).

Byte	Description

Byte	Description
0	Sub-data type (0x01)
1	Length – 18
2	Processor family identifier 0x01 – PIC18Fx520/620/720 0x02 – PIC18Fx621/525 0x03 – PIC18Fx527/622/627/722 (x=6/8) 0x04 – ARM Cortex M3 F10x 0x05 – ARM Cortex M3 L15x 0x07 – STM32F101RDT6 0x08 – STM32F103RFT6 0x09 – STM32F429IGH6 0x0A – STM32F103VET7 0x0B – STM32L151VDT6
3	Hardware interface and peripherals identifier 0x01 – 40/44 pin micro, peripherals as per family 0x02 – 64 pin micro, peripherals as per family 0x03 – 80 pin micro, peripherals as per family 0x04 – 64 pin STM32F101RCT6 0x05 – 64 pin STM32F103RDT6 0x06 – 64 pin STM32L151RDT6 0x07 – 176 pin micro, peripherals as per family
4-5	Size of program memory (in 1024 bytes units) (LSB)
	Size of program memory (in 1024 bytes units) (MSB)
6-7	Size of volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (LSB)
	Size of volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (MSB)
8-9	Size of internal non-volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (LSB)
	Size of internal non-volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (MSB)

Byte	Description
10-11	Size of external non-volatile memory (in 1024 bytes units) (LSB)
	Size of external non-volatile memory (in 1024 bytes units) (MSB)
12	External non-volatile memory type 0x01 – I2C generic NVM (most EEPROMs). 0x02 – SPI generic NVM. 0x03 – Adesto Rev. E 0x04 – SPI N25Q NVM 0x05 – SPI MX25L6433F
13	Hardware Version See Unit Hardware Version
14-15	Reprogramming facility identifier (LSB) Depends on HW/FW variant
	Reprogramming facility identifier (MSB) Depends on HW/FW variant
16-17	Script language version (LSB) (0x01)
	Script language version (MSB) (0x00)
18-19	Current Firmware ID (LSB) Note that this is in fact not a descriptor of the firmware platform per se, but rather a descriptor of the actual firmware running on the platform. However, it is a valuable field when a re-flash is considered.
	Current Firmware ID (MSB) Note that this is in fact not a descriptor of the firmware platform per se, but rather a descriptor of the actual firmware running on the platform. However, it is a valuable field when a re-flash is considered.

2.6.6 *Time and Location Stamp*

This sub-data is generated as a reply to Time and Location Stamp Request (0x04). It is also automatically added to the self-initiated messages generated by the unit.

Byte	Description

<table>
<tr><td>0</td><td>Sub-data type (0x04)</td></tr>
<tr><td>1</td><td>Length – 25</td></tr>
<tr><td>2</td><td>Location status (flags)
<table>
<tr><td>Time Accuracy

0 – Time is accurate
1 – Time is Inaccurate</td><td>GPS Connection

0 – Connected
1 – Not Connected</td><td colspan="6">Spare</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>3</td><td>Mode 1 (from GPS)</td></tr>
<tr><td>4</td><td>Mode 2 (from GPS)</td></tr>
<tr><td>5</td><td>Number of satellites used (from GPS)</td></tr>
<tr><td>6</td><td rowspan="4">Longitude</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td><td rowspan="4">Latitude</td></tr>
<tr><td>11</td></tr>
<tr><td>12</td></tr>
<tr><td>13</td></tr>
<tr><td>14</td><td rowspan="3">Altitude</td></tr>
<tr><td>15</td></tr>
<tr><td>16</td></tr>
<tr><td>17</td><td rowspan="2">Ground speed</td></tr>
<tr><td>18</td></tr>
<tr><td>19</td><td rowspan="2">Speed direction (true course)</td></tr>
<tr><td>20</td></tr>
<tr><td>21</td><td>UTC time – seconds</td></tr>
<tr><td>22</td><td>UTC time – minutes</td></tr>
</table>

23	UTC time – hours
24	UTC date – day
25	UTC date – month
26	UTC date – year Current Year minus 2000 (e.g. value of 7 = year 2007)

2.6.6.1 MODE 1/2 from GPS

Refer to Section 2.2.3.21

2.6.6.2 Number of Satellites Used

Refer to Section 2.2.3.22

2.6.6.3 Longitude, Latitude

Refer to Section 2.2.3.23

2.6.6.4 Altitude

Refer to Section 2.2.3.24

2.6.6.5 Ground Speed

This indicates the current speed (absolute value of the vector). It is represented as a 16-bit unsigned integer, in 10^{-2} meter/sec resolution (speed is represented in centimeters/second).

The source of speed data is either the GPS, the vehicle's CAN bus or frequency metering input as per unit's type, installation and configuration.

The reported value may monitor the immediate value of speed recorded upon generation of the message or the maximum value of speed from last report (as per the configuration). Byte 10, bit 6 of the message is monitoring the actual reported type.

2.6.6.6 Heading/Speed Direction (True Course)

Refer to Section 2.2.3.26

2.6.6.7 UTC Time

Refer to Section 2.2.3.27

2.6.6.8 UTC Date

Refer to Section 2.2.3.28

2.6.7 *Usage Counter*

This sub-data is generated as a reply to Usage Counter Request (0x07), or as a periodical update. In the latter case, it is sent with the Time and Location Stamp (sub-data 0x04).

Byte	Description
0	Sub-Data Type (0x07)
1	Length – 9
2	Spare
3	Counter 1 Input Number
4	Counter 1 Value (Minutes)
5	
6	
7	Counter 2 Input Number
8	Counter 2 Value (Minutes)
9	
10	

Input's Numbers Definition

2.6.8 *Authentication Table Update*

This sub-data is generated as a reply to Authentication Table Update Command (0x08).

Byte	Description	
0	Sub-Data Type (0x08)	
1	Length – 9	
2	Spare	
3	Authentication table Index 0	Authentication table Index 1
4	Authentication table Index 2	Authentication table Index 3

Byte	Description	
5	Authentication table Index 4	Authentication table Index 5
6	Authentication table Index 6	Authentication table Index 7
7	Authentication table Index 8	Authentication table Index 9
8	Authentication table Index 10	Authentication table Index 11
9	Authentication table Index 12	Authentication table Index 13
10	Authentication table Index 14	Authentication table Index 15

2.6.9 *Neighbor list of the Serving GSM Cell*

This sub-data is sent:

- Passively, as a reply to Cell ID Request (0x09).
- Actively, if enabled in unit's configuration, separately for home and roam GSM networks (addresses 201 and 203 respectively, bits 0, 1, 3 and 4).

Byte	Description
0	Sub-Data Type (0x09)
1	Length – 53
2	Spare (0x00)
3	seconds (0-59)
4	minutes (0-59)
5	hours (0-23)
6	day (1-31)
7	month (1-12)
8	Year (Current Year minus 2000 (e.g. value of 7 = year 2007))
9	Serving Cell BSIC (Base Station Identification Code)
10	Serving Cell LAC (LSB) (Localization Area Code)

Byte	Description
11	Serving Cell LAC (MSB) (Localization Area Code)
12	Serving Cell ID (LSB)
13	Serving Cell ID (MSB)
14	Serving Cell Power (Received signal strength in dBm (hex). The sign is not saved, this value is always representing a negative number)
15	Neighbor Cell 1 BSIC
16	Neighbor Cell 1 LAC (LSB)
17	Neighbor Cell 1 LAC (MSB)
18	Neighbor Cell 1 Cell ID (LSB)
19	Neighbor Cell 1 Cell ID (MSB)
20	Neighbor Cell 1 Power
…	…
45	Neighbor Cell 6 BSIC
46	Neighbor Cell 6 LAC (LSB)
47	Neighbor Cell 6 LAC (MSB)
48	Neighbor Cell 6 Cell ID (LSB)
49	Neighbor Cell 6 Cell ID (MSB)
50	Neighbor Cell 6 Power
	Zero Padding to complete the 55 bytes assigned for single event (if it's a logged event, i.e. sent actively)

2.6.10 *Maintenance Server Platform Manifest*

Periodically (or upon server command) the unit connects to a maintenance server in order to check for the latest firmware and/or programming update. Auto connection to the maintenance server can be enabled upon power up and upon firmware upgrade.

Upon connection the unit generates a sub-data which is described below.

If the unit cannot establish a connection to the maintenance server while the GPRS is available, it uses the dial up retry algorithm defined in the NVM Allocation (Anti-Flooding). If all the retries fail, the unit ceases to try and reconnects to an operational server (instead of entering Anti-Flooding, as it would do while connected to an operational server).

Byte	Description
0	Sub-data type (0x0A)
1	Length – 34
2	Processor family identifier 0x01 – PIC18Fx520/620/720 0x02 – PIC18Fx621/525 0x03 – PIC18Fx527/622/627/722 (x=6/8) 0x04 – ARM Cortex M3 F10x 0x05 – ARM Cortex M3 L15x 0x07 – STM32F101RDT6 0x08 – STM32F103RFT6 0x09 – STM32F429IGH6 0xA – STM32F103VET7 0x0B – STM32L151VDT6
3	Hardware interface and peripherals identifier 0x01 – 40/44 pin micro, peripherals as per family 0x02 – 64 pin micro, peripherals as per family 0x03 – 80 pin micro, peripherals as per family 0x04 – 64 pin STM32F101RCT6 0x05 – 64 pin STM32F103RDT6 0x06 – 64 pin STM32L151RDT6 0x07 – 176 pin micro, peripherals as per family
4-5	Size of program memory (in 1024 bytes units) (LSB)
	Size of program memory (in 1024 bytes units) (MSB)
6-7	Size of volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (LSB)
	Size of volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (MSB)

Byte	Description
8-9	Size of internal non-volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (LSB)
	Size of internal non-volatile memory (Divided by 128 bytes and rounded up/down to closest integer) (MSB)
10-11	Size of external non-volatile memory (in 1024 bytes units) (LSB)
	Size of external non-volatile memory (in 1024 bytes units) (MSB)
12	External non-volatile memory type 0x01 – I2C generic NVM (most EEPROMs). 0x02 – SPI generic NVM. 0x03 – Adesto Rev. E 0x04 – SPI N25Q NVM 0x05 – SPI MX25L6433F
13	Hardware Version See: Unit Hardware Version
14-15	Reprogramming facility identifier (LSB) Depends on HW/FW variant
	Reprogramming facility identifier (MSB) Depends on HW/FW variant
16-17	Script language version (LSB) (0x01)
	Script language version (MSB) (0x00)
18-19	Current Firmware ID (LSB) Note that this is in fact not a descriptor of the firmware platform per se, but rather a descriptor of the actual firmware running on the platform. However, it is a valuable field when a re-flash is considered.
	Current Firmware ID (MSB) Note that this is in fact not a descriptor of the firmware platform per se, but rather a descriptor of the actual firmware running on the platform. However, it is a valuable field when a re-flash is considered.
20-21	Current PL ID (LSB) Infrastructure only, currently not supported

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td></td><td>Current PL ID (MSB)
Infrastructure only, currently not supported</td></tr>
<tr><td>22-29</td><td>International mobile subscriber identity of the SIM (IMSI)
Reference to GSM 07.07, 15 chars maximum</td></tr>
<tr><td>30-32</td><td>Modem's firmware revision
From FW version 33x and later – 0x00
For FW versions older than 33x:
<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>30</td><td>Reserved (0)</td></tr>
<tr><td>31</td><td>Modem Revision ID, as presented in the table below</td></tr>
<tr><td>32</td><td>Modem Type Extension
(Extra byte, additional to the 3MSBits in the hardware byte of message type 0)</td></tr>
</table></td></tr>
<tr><td>33</td><td>Maintenance Configuration
<table>
<tr><td colspan="6">Spare</td><td>Firmware Upgrade Enabled
0 - Disabled
1 - Enabled</td><td>Programming Enabled
0 - Disabled
1 - Enabled</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>34</td><td rowspan="2">Release Candidate Revision ID
Little Endian 16 bit representing the Release Candidate SVN revision:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Formal Release</td></tr>
<tr><td>1-65535</td><td>The version is a Release Candidate. The 2 bytes represents the SVN revision number:
Example: If the hex file name is F000_...._RC540.hex the resulted
The binary value representing the decimal RC540 is 0x21C in little Endian.</td></tr>
</table></td></tr>
<tr><td>35</td></tr>
</table>

Modem Revision ID

ID (Dec)	Revision	Modem
0	Unknown	All (Used also in Nano from FW version 34d and on).
1	0.7.6	Enfora II
2	0.7.8	
3,4	reserved	
5	1.0.5	Enfora III
6	6.1.1 (Beta)	
7	1.1.1PKG30	
8	1.1.1PKG41	
9	D3-1.1.2PKG47	
10	D4-1.1.2PKG47	
11	D10.1.1.2	
12-20	reserved	
21	7.02.002	Telit II
22	7.02.100	
23	7.02.002	Telit III
24	7.02.003	
25	7.02.004	
26	7.03.000	
27	7.03.030 (Automotive)	
28	7.03.002	

ID (Dec)	Revision	Modem
29	7.03.032	
30	10.00.033 (Obsolete)	Telit V2
31	10.00.036	
32	10.00.035 (Obsolete)	
33	10.00.016	
34-40	reserved	
41	GLM-4-0610-000	Motorola 24L
42-50	Reserved for Motorola	
51	01.000	Cinterion BGS3
52	02.000	Nano: Cinterion BGS2-W.Rel2 (Used only up to FW version 34c).
53	03.001_arn00.000.14	Nano: Cinterion EHS5-E (Used only up to FW version 34c).
54	03.001_arn00.000.14	Nano: Cinterion EHS5-US (Used only up to FW version 34c).
55	03.001_arn00.000.14	Nano: Cinterion EHS6A (Used only up to FW version 34c).
56	03.001_arn01.000.08	Nano: Cinterion BGS2-W.Rel3

ID (Dec)	Revision	Modem
		(Used only up to FW version 34c).
57-70	Reserved for Cinterion	
71	12.00.002	Telit HE910-G (Reserved)
72	12.00.323	Telit HE910-NAD
73	13.00.003	Telit GE910 QUAD (V2)
74	12.00.504	Telit UE910-NAR
75	12.00.404	Telit UE910-EUR
76	10.00.023	Telit GE864 QUAD-V2
77	16.00.303	Telit GE910 QUAD-V3
78	10.00.027	Telit GE864 QUAD-V2
79	12.00.516	Telit UE910-NAD
80	12.00.416	Telit UE910-EUD
81	10.01.522	Telit GE864 QUAD-V2
82	12.00.506	Telit UE910-NAR (SSL)
83-255	Reserved	

2.6.11 *3G Cell ID Data*[5]

This sub-data is sent:

- Passively, as a reply to Cell ID data request (0x09). In this case the sub-data will be sent using the same communication transport as the request.
- Actively, if enabled in unit configuration, separately for home and roam GSM networks, on address 201 and 203 respectively, bits 0, 1, 3 and 4.

Byte	**Description**
0	Sub-Data Type (0x0C)
1	Length – 53
2	Spare

[5] Supported for 3G variants only

Byte	Description
3	seconds (0-59)
4	minutes (0-59)
5	hours (0-23)
6	day (1-31)
7	month (1-12)
8	Year (Current Year minus 2000 (e.g. value of 7 = year 2007))
9	MCC (LSB) (Mobile Country Code, Decimal, 200-901)
10	MCC (MSB) (Mobile Country Code, Decimal, 200-901)
11	MNC (LSB) (Mobile Network Code, Decimal, 0-999)
12	MNC (MSB) (Mobile Network Code, Decimal, 0-999)
13	LAC (LSB) (Localization Area Code)
14	LAC (MSB) (Localization Area Code)
15	RSCP (Received Signal Code Power - Energy after processing with gain from coding, which is equivalent to RSSI [RSCP=RSSI+EC/No]; dBm units, (-120)≤RSCP≤(-25))
16	Cell ID (Expanded 3G Cell ID (RNC + Cell ID + possible sector))
17	
18	
19	
20	Power (Received signal strength (hex); the sign is not saved, this value is always representing a negative number; dBm units)

Byte	Description
21	AcT (Access Technology) 0 – GSM 2 – UTRAN
22	PSC (LSB) (Primary Scrambling Code, Decimal, 0-65535)
23	PSC (MSB) (Primary Scrambling Code, Decimal, 0-65535)
…	Zero Padding to complete the 55 bytes assigned for single event (if it's a logged event, i.e. sent actively)

2.6.12 *Compressed Vector Change Report*

Note that this sub-data will NEVER be generated as real-time or distress events, only as a logged event.

If a corresponding functionality is enabled by configuration, the compressed vector change data is sent by the unit in the following cases:

- Upon detection of 6th vector change detection occurrence - in this case the system will generate a Message Type-9 containing all 6 vector change detection occurrences.
- Timeout - if at least one vector change event is stored in unit's memory and no other vector changes were generated by the unit during the pre-programmed period, the system will generate Message Type-9 containing all previous vector change detection occurrences.
- Upon Stop - Message Type-9 containing all previous vector change detection occurrences (if any) will be generated immediately upon stop report.
- Upon reset command - the Message Type-9 containing all previous vector change detection occurrences (if any) will be generated.

<table>
<tr><th>Byte</th><th colspan="8">Description</th></tr>
<tr><td>0</td><td colspan="8">Sub-Data Type (0x0D)</td></tr>
<tr><td>1</td><td colspan="8">Length – 53</td></tr>
<tr><td rowspan="3">2</td><td colspan="8">Number of included vector change detections</td></tr>
<tr><td colspan="5">Spare</td><td colspan="3">Number of included vector change detections</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>3-6</td><td colspan="5">Longitude</td><td colspan="3">Vector Change Detection 1</td></tr>
</table>

7-10	Latitude	
11-13	Odometer	
14	Spare	
15	Course	
16	Speed (km/h)	
17-19	Time	
20	Speed (km/h)	Vector Change Detection 2
21-22	Time from vector change (seconds)	
23-24	Delta Longitude (from last vector change)	
25-26	Delta Latitude (from last vector change)	
27-33	Vector Change Detection 3	
34-40	Vector Change Detection 4	
41-47	Vector Change Detection 5	
48-54	Vector Change Detection 6	

Each message will contain up to 6 vector change occurrences, while the first one is reported in its full format, and the rest are reported as a delta relative to the last point (see full message format on the next page).

Each vector change detection occurrence (except the first one) consumes 7 bytes containing a data of location change from the last vector change (or from the start event), time from the last event and speed.

The latitude, longitude and time of the first vector detection will be stored in its full format.

True course of the first location is reported as 8-bit unsigned integer. The conversion to degrees is according the equation below:

$$Course\ [degr] = \frac{Received\ value * 360}{255}$$

Possible values are 0 to 2п.

Timestamp of the first Vector change:

Minutes (LSB)	Seconds

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Hours (LSB)				Minutes (MSB)			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Spare		Days					Hours
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Delta Longitude and latitude (from last vector change) both are stored as signed integers, in 10^-8 radian resolution. Possible values are -Π to +Π for longitude, or –Π/2 to +Π/2 for latitude.

Time from last vector change is recorded in seconds.

The reported value of speed may monitor the immediate value of speed recorded upon generation of the message or the maximum value of speed from the last report (as per the configuration). Byte 10, bit 6 of the message is monitoring the actual reported type.

If there are less than 6 vector change detections in this message, the unit pads unused bytes of missing occurrences by zeros. The message length will remain constant.

2.6.13 *Modular Platform Manifest*

This sub-data is generated as a reply to Modular Platform Manifest Request (0x12).

Byte	Description
0	Sub-Data Type (0x12)
1	Length – Variable
2	Field 1 – Identifier
3	Field 1 – Length of Payload
4	Field 1 – Payload
…	…
…	Field N – Identifier
…	Field N – Length of Payload

Byte	Description
…	Field N – Payload

Fields Definition

Processor Family Identifier

Field ID – 0x0	0x00 – PIC18F6722 0x01 – STM32F101RCT6 0x02 – STM32F103RDT6 0x03 – STM32L151RDT6 0x04 – STM32F101RDT6 0x05 – STM32F103RFT6 0x06 – STM32F429IGH6 0x07 – STM32F103VET7 0x08 – STM32L151VDT6

Accelerometer Identifier

Field ID – 0x1	0x00 – MMA7260QT 0x01 – LIS331DL 0x02 – LIS331DLH (12 bit) 0x03 – LIS3DH (16 bit)

Size of Program Memory

Field ID – 0x2 (Kbytes)	Cello, CelloTrack-T: 256 (dec) Cello-IQ, Cello-CANiQ, CelloTrack Nano: 384 (dec)

Amount of Non-Volatile Memory Used by Application (e.g. configuration)

Field ID – 0x3 (Bytes)	Default 0 (N.A)

Size of Internal RAM

Field ID – 0x4 (Kbytes)	Cello, CelloTrack-T: 32 (dec) Cello-IQ, Cello-CANiQ: 64 (dec) CelloTrack Nano: 48 (dec)

Size of External Non-Volatile Memory

Field ID – 0x5	Cello, CelloTrack-T: 512(dec)

(Kbytes)	Cello-IQ, Cello-CANiQ: 8192(dec) CelloTrack Nano: 1024 (dec)

Amount of External Non-Volatile Memory Used by Application (e.g. configuration)

Field ID – 0x6 (Kbytes)	Cello, CelloTrack-T, CelloTrack Nano: 4 Cello-IQ, Cello-CANiQ: 8

Size of External RAM

Field ID – 0x7 (Bytes)	Default - 0 (N.A)

Current Firmware ID Number

Field ID – 0x8	Same as in Type-0 message

Current Hardware ID Number

Field ID – 0x9	Same as in Type-0 message. See new table here

Modem Type

Field ID – 0xA	Same as in Type-0 message. See new table here (only the 3 modem ID bits, for backwards compatibility)

Modem Firmware Version

Field ID – 0xB	Byte 2: Reserved (sent as zero) Byte 1: • 0, from FW version 33x and later • Per table below, for FW versions older than 33x Byte 0: Reserved (sent as zero)
	from FW version 33x and later: Modem firmware string returned from the Modem (Byte 1)
	…
	Modem firmware string returned from the Modem (Byte n)

Bytes 45-47 of Maintenance Platform Manifest contain the value of modem revision. The modem type is declared in a hardware byte; this field provides an additional definition.

Reserved (sent as zero)	Modem revision ID, as per table below	Reserved (sent as zero)
Byte 2	Byte 1	Byte 0

Modem revision ID: Refer to Modem Version ID

The new Modem firmware reporting mechanism is supported by the following products and FW versions:

- Cello-CAN(IQ) from FW version 33x and later
- CR300/B from FW version 43c and later
- CelloTrack Nano from FW 34d and later

GPS Type

Field ID – 0xC	00 – CEL3535 01 – CEL1500 02 – CEL1500L 03 – CEG-1000 (Internal) 04 – SIRF4 chip (internal) 05 – Glonass (internal) 06 – SIRF4 ROM – NMEA 07 – Telit JF2 (internal) 08 – Telit SE868-V2 (internal) 09 – Telit Modified JF2 (CelloTrack T) 10 – Telit SE868-V3 (internal) 16 – NMEA (CelloTrack T)

GPS Firmware

Field ID – 0xD	String as returned by GPS to revision request command

First Activation Date/Time

Field ID – 0xE

Byte	Description
0	Year
1	Month
2	Day
3	Second
4	Minute
5	Hour

Note that byte 0 is transmitted first, then byte 1 etc.
On the display it's shown as d/m/y h:m:s

FW Upgrade Date/Time

Field ID – 0xF	

<table>
<tr><td></td><td>
<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>0</td><td>Year</td></tr>
<tr><td>1</td><td>Month</td></tr>
<tr><td>2</td><td>Day</td></tr>
<tr><td>3</td><td>Second</td></tr>
<tr><td>4</td><td>Minute</td></tr>
<tr><td>5</td><td>Hour</td></tr>
</table>
Note that byte 0 is transmitted first, then byte 1 etc.
On the display it's shown as d/m/y h:m:s
</td></tr>
</table>

Last Configuration Change Date/Time

<table>
<tr><td>Field ID – 0x10</td><td>
<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>0</td><td>Year</td></tr>
<tr><td>1</td><td>Month</td></tr>
<tr><td>2</td><td>Day</td></tr>
<tr><td>3</td><td>Second</td></tr>
<tr><td>4</td><td>Minute</td></tr>
<tr><td>5</td><td>Hour</td></tr>
</table>
Note that byte 0 is transmitted first, then byte 1 etc.
On the display it's shown as d/m/y h:m:s
</td></tr>
</table>

Firmware File Name

Field ID – 0x11 (up to 120 bytes)	Firmware file name string

System ID (STM ID in case of STM controller)

Field ID – 0x12	12 bytes hexadecimal

Boot Loader ID

Field ID – 0x13	Contains 1 byte indicating Boot Loader's version number

DFD/SD Card Version

<table>
<tr><td>Field ID – 0x14</td><td>
<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>0</td><td>DFD Version Byte 0</td></tr>
<tr><td>1</td><td>DFD Version Byte 1</td></tr>
<tr><td>2</td><td>DFD Version Byte 2</td></tr>
<tr><td>3</td><td>DFD Version Byte 3</td></tr>
<tr><td>4</td><td>SD Card Version Byte 0</td></tr>
<tr><td>5</td><td>SD Card Version Byte 1</td></tr>
<tr><td>6</td><td>SD Card Version Byte 2</td></tr>
<tr><td>7</td><td>SD Card Version Byte 3</td></tr>
</table>
The SD card version is extracted from a file called ver.txt in the DFD's SD card root directory.
</td></tr>
</table>

Cello-CANiQ VIN

Field ID – 0x15	VIN – Vehicle Identification Number Null terminated string, Up to 17 Bytes

IMSI/IMEI/MEID

Field ID – 0x16	IMSI – 8 Bytes, decimal IMEI – 8 Bytes, decimal MEID – 8 Bytes, decimal (for CDMA devices)

Originating FW ID

<table>
<tr><td>Field ID – 0x17</td><td>This module holds the originating FW version or the last version the code tree was merged with.
<table><tr><th>Byte</th><th>Description</th></tr><tr><td>0</td><td>Originating Version ID</td></tr><tr><td>1</td><td>Originating Sub Version ID
The version of the trunk (Before Branching or after merging)</td></tr></table>Example: 33b
33 – Version ID
b – Sub Version → subversion Letter – 'a' = 'b'-'a'=1</td></tr>
</table>

Size of Internal Non-Volatile Memory

Field ID – 0x1A (Divided by 128 bytes and rounded up/down to closest integer)	Cello, CelloTrack-T, Cello-IQ, Cello-CANiQ: 0 (Dec) CelloTrack Nano: 96 (Dec) = 12KB

BT Module/Extender Identification

<table>
<tr><td>Field ID – 0x1B</td><td><table><tr><th>Byte</th><th>Description</th></tr><tr><td>0</td><td>Module ID
0 – CSR 1010</td></tr><tr><td>1</td><td>BLE stack version</td></tr><tr><td>2</td><td>CBLE application major version</td></tr><tr><td>3</td><td>CBLE application minor version</td></tr><tr><td>4-9</td><td>CBLE MAC address (MSB first)</td></tr></table></td></tr>
</table>

	<table><tr><th>Byte</th><th>Description</th></tr><tr><td>0</td><td>Module ID 1 – BlueGiga BT121</td></tr><tr><td>1</td><td rowspan="2">HW Revision</td></tr><tr><td>2</td></tr><tr><td>3</td><td rowspan="2">FW Version – Bootloader</td></tr><tr><td>4</td></tr><tr><td>5</td><td rowspan="2">FW Version – Major</td></tr><tr><td>6</td></tr><tr><td>7</td><td rowspan="2">FW Version – Minor</td></tr><tr><td>8</td></tr><tr><td>9</td><td rowspan="2">FW Version – Patch</td></tr><tr><td>10</td></tr><tr><td>11</td><td rowspan="2">FW Version – Build</td></tr><tr><td>12</td></tr><tr><td>13</td><td>Spare</td></tr></table>

SIM ICCID

Field ID – 0x1C	ASCII String (Up to 20 Bytes)

Modem Firmware Sub Version

Field ID – 0x1E	1 Byte, decimal

Maintenance Configuration

<table>
<tr><td rowspan="3">Field ID – 0x1F</td><td>Spare</td><td>Modem Firmware Upgrade Enabled
0 - Disabled
1 - Enabled</td><td>Firmware Upgrade Enabled
0 - Disabled
1 - Enabled</td><td>Programming Enabled
0 - Disabled
1 - Enabled</td></tr>
<tr><td>Bits 3-7</td><td>bit 2</td><td>bit 1</td><td>bit 0</td></tr>
</table>

2.6.14 *Pulse Counter Measurement Response*

This sub-data is generated as a reply to Pulse Counter Measurement Request (sub-data 0x14). It is sent with sub-data 0x04 (Time and Location Stamp).

Byte	Description
0	Sub-Data Type (0x14)
1	Length – 26
2	Spare
3	Spare
4 5 6 7	Counter 1 (Liter) 4 bytes forming unsigned 32 bits value representing the amount of litters consumed from the last pulse counter reset. The value is a multiplication of the pulse counter value by the scaling factor value (PL address 2442-2443 for Door input and 2444-2445 for Shock input).
8 9 10 11	Counter 2 (Liter) 4 bytes forming unsigned 32 bits value representing the amount of litters consumed from the last pulse counter reset. The value is a multiplication of the pulse counter value by the scaling factor value (PL address 2442-2443 for Door input and 2444-2445 for shock input).
12	Spare

Byte	Description
13	Spare
14	Spare
15	Spare
16	Spare
17	Spare
18	Spare
19	Spare
20	Spare
21	Spare
22	Spare
23	Spare
24	Spare
25	Spare
26	Spare
27	Spare

NOTE: Litters are only one example for volume measurement units. Actually the real measurement units are defined by the measuring device and its fuel volume vs. pulses relation.

2.6.15 *One-Wire Temperature Sensor Measurement*

This sub-data holds the One-Wire temperature sensor measurements. This message is generated by the unit as a reply to One-Wire Temperature Sensor Measurement Request (sub-data 0x19). It is sent with sub-data 0x04 (Time and Location Stamp).

Byte	Description

Byte	Description
0	Sub-Data Type (0x19)
1	Length – 26
2	First One-Wire ID (Byte 0)
3	First One-Wire ID (Byte 1)
4	First One-Wire ID (Byte 2)
5	First One-Wire ID (Byte 3)
6	First One-Wire measurement (LSB) (Coefficient 0.0625)
7	First One-Wire measurement (MSB) (Coefficient 0.0625)
8	Second One-Wire ID (Byte 0)
9	Second One-Wire ID (Byte 1)
10	Second One-Wire ID (Byte 2)
11	Second One-Wire ID (Byte 3)
12	Second One-Wire measurement (LSB) (Coefficient 0.0625)
13	Second One-Wire measurement (MSB) (Coefficient 0.0625)
14	Third One-Wire ID (Byte 0)
15	Third One-Wire ID (Byte 1)
16	Third One-Wire ID (Byte 2)
17	Third One-Wire ID (Byte 3)
18	Third One-Wire measurement (LSB) (Coefficient 0.0625)
19	Third One-Wire measurement (MSB) (Coefficient 0.0625)
20	Fourth One-Wire ID (Byte 0)
21	Fourth One-Wire ID (Byte 1)

Byte	Description
22	Fourth One-Wire ID (Byte 2)
23	Fourth One-Wire ID (Byte 3)
24	Fourth One-Wire measurement (LSB) (Coefficient 0.0625)
25	Fourth One-Wire measurement (MSB) (Coefficient 0.0625)
26	Spare
27	Spare

2.6.16 *Car Sharing 2 Reservation Entry Response*

This sub-data is sent as a reply to Car Sharing 2 Reservation Command message (0x1A) with Read command from server.

Byte	Description
0	Sub-Data Type (0x1A)
1	Length – 53
2	Slot Number
3	Spare
4–9	Driver ID (SCN) Bytes 0-5 of reservation table entry
10-13	Reservation Start time/date Bytes 6-9 of reservation table entry (Number of Seconds from December 31, 1989, 12 am UTC.)
14-48	Spare - Zero Padded

2.6.17 *CDMA Cell ID Data*[6]

This sub-data will be sent:

- Passively, as a response to Cell ID data request (0x09). In this case the packet will be sent using the same communication transport as the request.
- Actively, if enabled in unit's configuration, separately for home and roam GSM networks, on addresses 201 and 203 respectively, bits 3 and 4.

Byte	Description
0	Sub-Data Type (0x1C)
1	Length (0x35)
2	Spare
3	seconds (0-59)
4	minutes (0-59)
5	hours (0-23)
6	day (1-31)
7	month (1-12)
8	Year (Current Year minus 2000 (e.g. value of 7 = year 2007))
9	SID (LSB) (System ID, Decimal, 0-999)
10	SID (MSB) (System ID, Decimal, 0-999)
11	NID (LSB) (Network ID)
12	NID (MSB) (Network ID)
13	RSSI (Received Signal Strength Index [RSSI=Rx Power + EC/Io]; dBm units)
14	BSID (Base Station ID (Cell ID + possible sector))
15	
16	

[6] Supported for CDMA variants only

Byte	Description
17	
…	Zero Padding to complete the 55 bytes assigned for single event (if it's a logged event, i.e. sent actively)

2.7 Modular Message (Message Type 11)

Type 11 was introduced for supporting true modular protocol. The basic structure of the protocol is designed to carry records with predefined structure called modules. The protocol will be used as an extension for Cello fleet protocol. Type 11 supports theoretical message length of up to 65536 bytes, tough the actual rate will be constrained by the HW limitations.

2.7.1 *Message Ingredients*

Type 11 contains the following data (listed in the actual transmitted order):

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Communication Control Field – 2 bytes
  - Message Numerator – 1 byte
- Packet Control Field – Legacy fleet field
- Message length – 2 bytes
- Spare – 4 bytes
- Payload Modules – User Configuration Depended
- Error Detection Code (checksum) – 1 byte

2.7.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”
4	System Code, byte 4 – ASCII “P”
5	Message Type (11)
6	Unit ID
7	
8	
9	
10	Communication Control Field

11	
12	Message Numerator
13	Packet Control Field
14	Length of the modules section (From byte #16 and not including the last byte of the checksum)
15	
16	= 0x0000
17	Symbolizes outbound message (while in inbound these 2 bytes are allocated to length which is ≠ 0)
18	Spare (sent as 0)
19	
20-28	Module Name 8 - FW_HW ID (Mandatory)
29-50	Module Name 6 - GPS Location Stamp (Mandatory)
51-60	Module Name 7 - Time stamp (Mandatory)
…	
	Other Modules
…	
…	
Last Byte	Error Detection Code (8-bit additive checksum, excluding system code)

2.7.3 *Detailed Per-Field Specifications*

2.7.3.1 System Code

Refer to Section 2.2.3.1

2.7.3.2 Message Type

Modular messages contain a value of 11 (eleven) in the message type field.

2.7.3.3 Unit ID

Refer to Section 2.2.3.3

2.7.3.4 Communication Control Field

Refer to Section 2.2.3.4

2.7.3.5 Command Numerator (Anti-Tango™)

Refer to Section 2.2.3.5

2.7.3.6 Packet Control Field

Direction	Out of space indication	Unused					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Direction

0 – Data from the unit

1 – Request (unit-bound)

Out of Space Indication

0 – All the requested data is present in the message.

1 – Some Sub-data was not returned due to data size.

2.7.3.7 Length

That field includes the number of data bytes of the modules (with their codes and lengths). It is the number of bytes from byte 16 to the byte of the checksum, which is not included.

2.7.4 *Outbound Type 11 Module Structure*

The general structure of a type 11 module is as follows:

Byte	Description
0	Module Code
1	Length of module – Number of payload bytes
2	
3	Module Payload bytes
.	
.	
n	

2.7.5 *Outbound Type 11 Modules Table*

Code	Description
6	GPS Location Stamp
7	GPS Time Stamp

Code	Description
8	FW ID
9	ACK/NACK
10	Configuration Memory Write Response
11	Configuration Memory Read Response
13	Authenticated Features Query Response
16	Modem FOTA Response
30	Reserved for PointerCept General Status Event

2.7.6 *GPS Location Stamp*

Byte	Description
0	Module 6 - GPS Location Stamp
1	Length of module - 19
2	
3	HDOP
4	Mode 1 (from GPS)
5	Mode 2 (from GPS)
6	Number of satellites used (from GPS)
7	Longitude
8	
9	
10	
11	Latitude
12	
13	
14	
15	Altitude
16	
17	

18	
19	Ground speed (km/h)
20	Speed direction (true course)
21	

Refer to Sections 2.2.3.21 until 2.2.3.26 for more details about fields' data formats.

2.7.7 *GPS Time Stamp*

Byte	Description
0	Module 7 – GPS Time Stamp
1	Length of module - 7
2	
3	Validity of time / GPS Fix (valid - 1 /invalid - 0)
4	System time – seconds
5	System time – minutes
6	System time – hours
7	System date – day
8	System date – month
9	System date – year (-2000)

2.7.8 *Firmware ID*

Byte	Description
0	Module 8 - FW ID
1	Length of module - 6
2	
3	Bits 0-3: Active cellular technology 0 – Unknown 1 – Reserved 2 – 2G 3 – 3G 4 – 4G (LTE CAT-1) 5 - LTE CAT-M

	6-15 - Reserved Bits 4-5: Hub kind 0 – Cellocator hub 1 – Android based hub 2 – iOS based hub 3 - Reserved Bits 6-7: Spare
4	Type 11 Protocol ID (=1)
5	FW Version ID (example: 33)
6	FW Sub-Version ID (example: 1 for a)
7	HW ID (example: 20)
8	Spare

2.7.9 *ACK/NACK*

Byte	Description
0	Module 9 – ACK/NACK
1	Length of module – 3
2	
3	0 - ACK 1 - NACK
4	NACK Code (decimal) 0 - General NACK 70 - Exceeded Number of Failed Feature Authentication Attempts (the unit will ignore Feature Authentication command for the next hour) 71 - Feature Authentication Code Discrepancy 90 - Modem FOTA Process Cannot Start due to Operation on Internal Battery 91 - Modem FOTA Process with Maintenance Server Cannot Start because Disabled in PL 92 - Modem FOTA Process Cannot Start due to Ongoing FW Upgrade Process 93 - Modem FOTA Process Cannot Start due to Ongoing PL Upgrade Process 94 - Modem FOTA Process Cannot Start due to Ongoing Modem FOTA

	Upgrade Process
5	Spare

2.7.10 *Configuration Memory Write Response*

Byte	Description
0	Module 10 – Configuration Memory Write Response
1	Length of module – Variable
2	
3	Numerator
4	
5	Number of instances ACK
6	Instance 1 action status 0 - OK 1 - Write Error
7	Instance 2 action status 0 - OK 1 - Write Error
	…

2.7.11 *Configuration Memory Read Response*

Byte	Description	
0	Module 11 – Configuration Memory Read Response	
1	Length of module – Variable	
2		
3	Numerator	
4		
5	Number of Instances	
6	Memory type – 0	
7	Memory entry unit type	Instance 1

	0 – Bit 1 – Byte 2 – Word (16 bits) 3 – Double Word (32 bits) (Only Byte entry unit type is currently supported)	
8	Address in the configuration memory space	
9		
10		
11		
12	Number of Entries	
13		
…	Data Payload	
…	…	Instance 2
…	…	
…	…	

2.7.12 *Authenticated Features Query Response*

This module enables sending features bitmaps upon receiving Authenticated Features Query Command (module 13).

This module shall be sent with mandatory module 8 (FW ID).

Byte	Description
0	Module 13 – Authenticated Features Query Response
1	Length of module – 21
2	
3	Spare
4	Spare
5	Authenticated Features Matrix Byte 0 <table><tr><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td></tr><tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr></table>

<table>
<tr><td>6</td><td>Authenticated Features Matrix Byte 1
<table>
<tr><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>Obs.</td><td>PointerCept
0 - Inactive
1 - Active</td><td>Obs.</td><td>Obs.</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>7</td><td>Authenticated Features Matrix Byte 2
<table>
<tr><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Basic Driver Behavior
0 - Inactive
1 - Active</td><td>TDLT
0 - Inactive
1 - Active</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table></td></tr>
<tr><td>8</td><td>Authenticated Features Matrix Byte 3</td></tr>
<tr><td>9</td><td>Authenticated Features Matrix Byte 4</td></tr>
<tr><td>10</td><td>Authenticated Features Matrix Byte 5</td></tr>
<tr><td>11</td><td>Authenticated Features Matrix Byte 6</td></tr>
<tr><td>12</td><td>Authenticated Features Matrix Byte 7</td></tr>
<tr><td>13</td><td>Authenticated Features Matrix Byte 8</td></tr>
<tr><td>14</td><td>Authenticated Features Matrix Byte 9</td></tr>
<tr><td>15</td><td>Authenticated Features Matrix Byte 10</td></tr>
<tr><td>16</td><td>Authenticated Features Matrix Byte 11</td></tr>
<tr><td>17</td><td>Authenticated Features Matrix Byte 12</td></tr>
<tr><td>18</td><td>Authenticated Features Matrix Byte 13</td></tr>
<tr><td>19</td><td>Authenticated Features Matrix Byte 14</td></tr>
<tr><td>20</td><td>Spare</td></tr>
<tr><td>21</td><td>Spare</td></tr>
<tr><td>22</td><td>Spare</td></tr>
<tr><td>23</td><td>Spare</td></tr>
</table>

2.7.13 *Modem FOTA Response*

This module enables sending Modem FOTA process results, upon finish of Modem FOTA process, initiated by receiving Modem FOTA Command (module 16).

Byte	Description
0	Module 16 – Modem FOTA Response
1	Length of module – Variable
2	
3	Spare
4	Spare
5	Spare
6	Modem FOTA Process Status 0 – ACK (Process Ended Successfully) 1 – Modem Nack, General Error 2 – Modem Nack, Firmware Corrupted, CRC Error 3 – Modem Nack, Firmware Package Mismatch 4 – Modem Nack, Firmware Signature Failed 5 – Modem Nack, Authentication Failed 6 – Modem Nack, Out of Memory Resource 20 – FW (MCU) Nack, FTP Session Failed 21 – FW (MCU) Nack, Illegal FTP Directory or Non-Exist Directory 22 – FW (MCU) Nack, File Downloading Failed 23 – FW (MCU) Nack, Upgrade Failed 24 – FW (MCU) Nack, Upgrade Timeout Expired 30 – FW (MCU) Nack, General Error 31 – FW (MCU) Nack, FOTA Process Terminated due to Unit Reset
7	Spare
8	Spare
9	Modem Sub Version After Upgrade Attempt (1 byte, 0-255)
…	Modem Version After Upgrade Attempt (ASCII String, 1st byte is length)

3 Command Channel (Inbound Messages)

3.1 Overview

The command channel comprises several types of messages, as described in the following:

- **Generic Command (Message Type 0)** – some commands are sent using this legacy message. This message is always replied with a legacy status/location message from the target unit (if the command is received successfully).
- **Programming Command (Message Type 1)** – this message provides OTA programming capabilities, and is always replied to with a programming data message from the target unit, when received correctly.
- **Generic Acknowledge Message (Message Type 4)** – this message is sent by the server to verify reception of outbound status/location, telemetry or transparent data messages.
- **Forward Data Command (Message Type 5)** – this message allows the sending of data to the terminal attached to the unit.
- **Modular Message Request (Message Type 9)** – this legacy modular message is designed to request the unit to send types of data, defined in Modular Message packet like CAN bus sensors, Cell ID, debug data, etc.
- **Self Re-flash Chunks (Message Type 10)** – this message forwards firmware file data chunks for the self-re-flash process of the unit. The Self re-flash process description is outside of the scope of the current document.
- **Modular Message Request (Message Type 11)** – this modular message type implements an extended modular protocol, intended to replace older message types (0, 1, and 9). It is used to request the unit to send many types of data in a modular message packet, like CAN bus applications, CelloTrack Nano, etc.

3.2 Generic Command (Message Type 0)

The generic command message is the main command interface to the unit.

3.2.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Command Numerator – 1 byte
  - Authentication Code – 4 bytes
- Command Code – 1 byte (repeated twice)
- 1st Command Data Field – 1 byte (repeated twice)
- 2nd Command Data Field – 1 byte (repeated twice)
- Command Specific Data Field – 4 bytes
- Error Detection Code – 1 byte

3.2.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”
4	System Code, byte 4 – ASCII “P”
5	Message Type (0)
6	Unit ID
7	
8	
9	
10	Command Numerator
11	Authentication Code
12	
13	
14	

15	Command Code
16	Command Code (repetition)
17	1st Command Data Field
18	1st Command Data Field (repetition)
19	2nd Command Data Field
20	2nd Command Data Field (repetition)
21	Command Specific Data Field
22	
23	
24	
25	Error Detection Code (8-bit additive checksum, excluding system code)

3.2.3 *Detailed Per-Field Specifications*

3.2.3.1 System Code

The same system code constant that is sent on every message – ASCII “M”, “C”, “G”, “P” or “M”, “C”, “G”, “S”, in this order.

3.2.3.2 Message Type

Message type identifies the kind of the message. It allows the receiver to differentiate between different messages types, according to the value sent in this field.

Generic command messages contain a value of 0 (zero) in the message type field.

3.2.3.3 Unit ID

This field contains the unique unit ID of the target unit. The unit ignores all received commands that do not contain the appropriate unit ID number.

3.2.3.4 Command Numerator Field

This field should contain the number of the command. This number appears in the "Message numerator" field in the unit reply message, enabling the user to easily distinguish between acknowledged commands and un-acknowledged ones.

3.2.3.5 Authentication Code

This field contains a 4 byte unique authentication code, which is verified by the unit, in order to provide protection against unapproved command attempts (from FW 27p and up). For example: an attempt to change the traffic destination IP by unauthorized personnel.

If the code is not verified as authentic – the unit will not perform/acknowledge the command.

The feature should be switched on in the unit configuration (refer to Programming Manual for more details).The feature is switched off by default.

The 4 bytes authentication code is generated as a function of two variables:

- Unit ID
- 8 bytes Authentication Table, stored in the NVM of the unit and concurrently in the Communication Center application (refer to Modular Message Definition for modification instructions to this table).

The OTA Authentication table modification will be only accepted by the unit if the Command Authentication feature is **DISABLED**.

The following are default values of the Authentication table:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Value	2	15	7	9	12	1	4	6	8	3	11	14	0	5	10	13

3.2.3.6 Command Code

As the generic command message is relevant for all kinds of commands, it is necessary to specify the actual command that is desired. Therefore, each different command assigns a unique command code, which is used in the command code field, to specify the command to be executed.

3.2.3.7 Command Data Fields (1st and 2nd)

The command data fields contain further information, which is needed by some of the commands.

3.2.3.8 Command Specific Data Field

The command specific data field contains additional information, which is needed by some of the commands.

The available commands and corresponding data fields are detailed below:

Command Code (Hex)	Description
0x00	Immediate status request
0x02	Unit state change Data field value: 0x00: Go to Standby 0x01: Go to Emergency mode This command sets the unit to start transmitting emergency messages

Command Code (Hex)	Description
	according to the command configuration. The command is sent with two parameters, the interval between each emergency transmission and how many transmissions to send to the operator. If the number of transmissions chosen is 0, the unit sends the emergency transmission constantly. If the time between transmissions is set to 0, the unit sends the emergency transmission according to the pre-programmed definition of the Distress Mode in the NVM. The emergency command is meant to emulate the action of a driver pressing on the emergency button. It uses the same mechanism. If an emergency command is sent and the driver simultaneously presses on the emergency button, the emergency function that the driver initiated stops the command sent by the operator and starts its own emergency session. Here is an example of the emergency command sent to a unit: Number of distress transmissions = 2 Time between distress transmissions Events = 5sec 4D 43 47 50 00 4B 01 00 00 1C 6E DF DD DD 02 02 01 01 00 00 02 05 00 00 7C 0x02: Reset The following fields will be reset: The "Garmin Enabled", "Garmin Connected" and GSM hibernation indication bit flags, Message numerator, Unit's status, Current GSM operator report, Unit's mode of operation, I/O, Analog inputs, Driver ID /PSP Specific Data/Accelerometer Status, Last GPS Fix, Number of satellites, Longitude, Latitude, Altitude, Speed, Course, System time, System date. The modem will be re-initialized, the GPRS connection restored. The RAM buffer used for data forwarding will be reset. Configuration parameters will be reloaded from Configuration memory. Command Specific Data field: don't care
0x03	Output state change Data field should contain output change information, according to this table: Data field 1 value: function 00h / 10h: Siren (off / on) 01h / 11h: Hood lock (off / on), in 370-x0 only 02h / 12h: SP1W (off / on,) in 370-x0 only 03h / 13h: Ext Immobilizer (Same output as Gradual Stop) (off / on)

Command Code (Hex)	Description
	04h / 14h: Blinkers (off / on) 05h / 15h: Standard immobilizer 1 (off / on) 06h / 16h: Speaker phone voltage (off / on), in 370-x0 only 07h / 17h: Internal lights (off / on), in 370-x0 only 08h / 18h: LED (off / on), in 370-x0 only 09h / 19h: General Output (off / on), in 370-x0 only 0Ah / 1Ah: Windows (off / on), in 370-x0 only 0Bh / 1Bh: Stop Light (off / on), in 370-x0 only 0Ch / 1Ch: Buzzer (off / on), in 370-x0 only 0Eh: Lock (performs pulse), in 370-x0 and Olympic modifications only 0Fh: Unlock (performs pulse),), in 370-x0 and Olympic modifications only Data field 2 and 2 bytes of Command Specific Data field: Contain time of the output activation with one second resolution. Value of 0 cause permanent output change. Example: Activate Siren for 5 minutes (300 seconds). MCGP 00 ID ID ID ID 00 00 00 00 00 03 03 10 10 2C 2C 01 01 00 00 CS Nested output activation: If the MSBit of the 3rd byte of command specific data field is set, the command will be executed only after the vehicle stops, e.g. after Ignition off or after 10 (by default) valid GPS packets showing speed lower than 1 km/h). Example: Activate Siren Nested for 5 minutes (300 seconds). MCGP 00 ID ID ID ID 00 00 00 00 00 03 03 10 10 2C 2C 01 01 80 00 CS
0x04	Disable Active Transmissions. This command will control the corresponding bit in the unit's configuration (address 6, bit 1) and immediately stop or restore active transmissions generated by the end unit. The existing GPRS session will be disconnected upon "disable command" or restored upon "Enable command". Data field: 0 – Disable active transmissions 1 – Enable active transmissions Command Specific Data field: don't care
0x05	Tracking control command (based on time events). Data field: zero to stop tracking, non-zero sets the resolution of time events and immediately implements it. Refer to Programming Manual for values. Command Specific Data field: don't care

<table>
<tr><th>Command Code (Hex)</th><th>Description</th></tr>
<tr><td>0x07</td><td>Commence gradual engine stop (PWM Immobilizer - from 100% to 0% duty cycle). Data field must contain zero (a non-zero value stops Immobilizer).
Command Specific Data field: don't care</td></tr>
<tr><td>0x0D</td><td>Erase tracking Log from NVM memory
Data field: don't care</td></tr>
<tr><td>0x0E</td><td>Reset GPS receiver
Data field:
Zero for standard reset (by On/Off pin)
1st = 0x5A
2nd =0xA5
For Factory GPS reset command. Note, that the unit can (configurable) perform GPS reset automatically in the following cases:
Standard reset (by On/Off pin) on ignition off.
If the GPS is communicating, but not navigating and MODE1=0, MODE2=16 for 10 minutes the unit performs a factory GPS reset.
If the GPS is not communicating, or communicating but not navigating and MODE1≠0, MODE2≠16 for 15 minutes the unit performs standard GPS reset.
If same condition as in item 3 remains true for the next 15 minutes the unit performs a factory GPS reset.</td></tr>
<tr><td>0x10</td><td>Force GPS energizing (Not supported by Cello family)
The command allows maintaining GPS activated, regardless of hibernation logic.
Warning: Note that only GPS is affected by this command! If GPS is forced active, there is no way to send a command to revert the GPS back to automatic behavior while communication is down (due to the hibernation mask or due to shutdown of the modem as a result of the full hibernation).
1st + 2nd command data fields:
A value of 1 (one) to force energizing of GPS.
A value of 0 (zero) for automatic GPS behavior (according to normal logic).</td></tr>
<tr><td>0x12</td><td>Connect to server (from FW28 and up)
0 – Main server
1 – Secondary server (provisioning)</td></tr>
</table>

<table>
<tr><th>Command Code (Hex)</th><th>Description</th></tr>
<tr><td></td><td>2 – Maintenance Server</td></tr>
<tr><td>0x14</td><td>Calibrate frequency counters
Data field 1 contains description of the calibration type:
<table>
<tr><td colspan="6">Reserved</td><td>Source type
0 – GP Freq. (RPM)
1 – Speed</td><td>Calibrated input
0 – pin 14
1 – pin 15</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
Data field 2:
• In case of GP Frequency contains percent of maximum engine load (i.e 10 for 10%)
• In case of speed – required distance in hundred's meters (recommended value 5km).
Command Specific Data field: don't care</td></tr>
<tr><td>0x15</td><td>Control of transparent mode over COM (independent from control by Door input (pin 14)).
Data field 1 contains action code: 0 - deactivate, 1- activate
Data field 2 contains activation time (in seconds); 1 to 255 seconds, 0 - permanent activation
If activation by Door input (pin 14) is enabled: The OTA command overwrites input setting.</td></tr>
<tr><td>0x16</td><td>Query connected trailer ID
Data field: don’t care</td></tr>
<tr><td>0x1B</td><td>PointerCept Control Command (Infrastructure)
Data field 1 contains action code: 0 - deactivate, 1- activate
Data field 2 contains activation time (in 10 minutes units): 1 to 255 minutes, 0 – illegal value</td></tr>
</table>

3.3 Programming Command (Message Type 1)

The programming command message allows to configure the unit.

NOTE: For configuration spaces larger than 4K (Typically in Cello-IQ and Cello-CANiQ units) it is mandatory to use Type 11 programming command (modules 10, 11).

3.3.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Command Numerator – 1 byte
  - Authentication Code – 4 bytes
- Block Code – 1 byte
- Programming Masking Bitmap – 2 bytes
- Block Data – 16 bytes
- Error Detection Code – 1 byte

3.3.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type (1)
6	Unit ID
7	
8	
9	
10	Command Numerator
11	Authentication Code
12	

<table>
<tr><td>13</td><td rowspan="2"></td></tr>
<tr><td>14</td></tr>
<tr><td>15</td><td>Block Code</td></tr>
<tr><td>16</td><td>Programming Masking Bitmap</td></tr>
<tr><td>17</td><td><table><tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="8">Byte 17</td><td colspan="8">Byte 16</td></tr></table></td></tr>
<tr><td>18-33</td><td>Block Data</td></tr>
<tr><td>34</td><td>Error Detection Code (8-bit additive checksum, excluding system code)</td></tr>
</table>

3.3.3 *Detailed Per-Field Specifications*

3.3.3.1 System Code

Refer to Section 3.2.3.1

3.3.3.2 Message Type

Programming Command messages contain a value of 1 (one) in the message type field.

3.3.3.3 Unit ID

Refer to section 3.3.3.33.2.3.3

3.3.3.4 Command Numerator Field

Refer to section 3.2.3.4

3.3.3.5 Authentication Code

Refer to section 3.2.3.5

3.3.3.6 Block Code

OTA (over the air) parameter programming is done in blocks. The entire NVM parameter memory is partitioned to 16-bytes long blocks. Each of those blocks is uniquely identified with a block code. The block code field contains the code of the block whose data is sent in this message (in the block data field).

3.3.3.7 Programming Masking Bitmap

The bitmap allows programming of only part of the parameters in a block, while leaving the other parameters with their previous values.

Each bit in the 16-bit value represents a byte in the parameters memory block. A value of "1" in a certain bit enables programming to the corresponding byte in the parameters memory, where a value of "0" prohibits programming of that byte.

3.3.3.8 Block Data

Contains the actual data programmed in the specified block of the parameter memory.

3.4 Generic Acknowledge Message (Message Type 4)

The generic acknowledge message is an inbound message sent by server to verify reception of outbound Status/Location (Type 0), Data Forwarding (Type 7, 8) and Modular (Type 9) messages.

3.4.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Command Numerator – 1 byte
  - Authentication Code – 4 bytes
- Action Code – 1 byte
- Main Acknowledge Number – 2 bytes (1 reserved)
- Secondary Acknowledge Number – 2 bytes (reserved)
- Compressed Date – 2 bytes
- Compressed Time – 2 bytes
- Spare – 2 bytes
- Error Detection Code – 1 byte

3.4.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”
4	System Code, byte 4 – ASCII “P”
5	Message Type (4)
6	Unit ID
7	
8	
9	
10	Command Numerator
11	Authentication Code

<table>
<tr><td>12</td><td rowspan="3"></td></tr>
<tr><td>13</td></tr>
<tr><td>14</td></tr>
<tr><td>15</td><td>Action Code (sent as zero)</td></tr>
<tr><td>16</td><td>Main Acknowledge Number – LSB</td></tr>
<tr><td>17</td><td>Reserved for Main Acknowledge Number – MSB (sent as zeros)</td></tr>
<tr><td>18</td><td>Reserved for Secondary Acknowledge Number – LSB (sent as zeros)</td></tr>
<tr><td>19</td><td>Reserved for Secondary Acknowledge Number – MSB (sent as zeros)</td></tr>
<tr><td>20</td><td>Reserved for future use (sent as zeros)</td></tr>
<tr><td>21
22</td><td>Compressed Date
<table>
<tr><td colspan="5">Day</td><td colspan="4">Month</td><td colspan="7">Year (-2000)</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">Byte 22</td><td colspan="8">Byte 21</td></tr>
</table></td></tr>
<tr><td>23
24
25</td><td>Compressed Time
<table>
<tr><td colspan="7">Spare (sent as 128)</td><td colspan="6">Seconds</td><td colspan="6">Minutes</td><td colspan="5">Hours</td></tr>
<tr><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">Byte 25</td><td colspan="8">Byte 24</td><td colspan="8">Byte 23</td></tr>
</table></td></tr>
<tr><td>26
27</td><td>Spare</td></tr>
<tr><td>28</td><td>Error Detection Code (8-bit additive checksum, excluding system code)</td></tr>
</table>

3.4.3 *Detailed Per-Field Specifications*

3.4.3.1 System Code

Refer to Section 3.2.3.1

3.4.3.2 Message Type

Generic Acknowledge messages contain a value of 4 (four) in the message type field.

3.4.3.3 Unit ID

Refer to section 3.2.3.3

3.4.3.4 Command Numerator Field

Refer to section 3.2.3.4

3.4.3.5 Authentication Code

Refer to section 3.2.3.5

3.4.3.6 Action Code

Sent as zero.

3.4.3.7 Main Acknowledge Number

This field contains the Message Numerator filed of the acknowledged outbound message.

3.4.3.8 Secondary Acknowledge Number

Currently not used and sent as zero.

3.5 Forward Data Command (Message Type 5)

The unit can forward data from the OTA channel to its serial port.

3.5.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Command Numerator – 1 byte
  - Authentication Code – 4 bytes
- Settings Byte – 1 byte
- Data length – 1 byte + 2 bits of Settings Byte (the 2 bits extension is applicable only for transparent mode)
- Data to Forward – variable length (up to 255 bytes (in regular forwarding mode) or up to 518 bytes (in transparent mode))
- Error Detection Code – 1 byte

3.5.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”
2	System Code, byte 2 – ASCII “C”
3	System Code, byte 3 – ASCII “G”
4	System Code, byte 4 – ASCII “P”
5	Message Type (5)
6	Unit ID
7	
8	
9	
10	Command Numerator
11	Authentication Code
12	
13	
14	

<table>
<tr><td>15</td><td>Settings Byte
<table>
<tr><td colspan="3">Destination of Payload
0 – N/A
1 – N/A
2 – COM2 (BT)
3 – COM3
4 – COM4
5 – COM5
6 – CFE Micro
7 – N/A</td><td colspan="2">Reserved</td><td>Data Length Bit 9 (MSB)</td><td>Data Length (Bit 8)</td><td>Packet to Garmin (Garmin serial protocol)
0 – Not to Garmin
1 – To Garmin</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
</table>
</td></tr>
<tr><td>16</td><td>Data Length</td></tr>
<tr><td>17</td><td>Data to Forward</td></tr>
</table>

3.5.3 *Detailed Per-Field Specifications*

3.5.3.1 System Code

Refer to Section 3.2.3.1

3.5.3.2 Message Type

Forward Data Command messages contain a value of 5 (five) in the message type field.

3.5.3.3 Unit ID

Refer to section 3.3.3.33.2.3.3

3.5.3.4 Command Numerator Field

Refer to section 3.2.3.4

3.5.3.5 Authentication Code

Refer to section 3.2.3.5

3.5.3.6 Settings Byte

This byte is used for different system indications.

Data Length Bits 8 and 9 extension will be used for transparent mode only.

In case CFE configuration is used, Destination of Payload will define the target CFE serial port.

3.5.3.7 Data Length

This field should contain a number of bytes to forward: up to 255 bytes in regular forwarding mode, or 511 bytes in transparent mode.

3.5.3.8 Data to Forward

This is the data that is forwarded to the serial port. This field must be an exact number of bytes long, as specified in the Data Length field.

3.6 Modular Message Request (Message Type 9)

The modular data packet request is designed to provide different data types in the same packet.

3.6.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Unit ID – 4 bytes
  - Command Numerator – 1 byte
- Authentication Code – 4 bytes
- Packet Control Field – 1 byte
- Message Length – 1 byte
- First Sub-Data Type – 1 byte
- First Sub-Data Length – 1 byte
- First Sub-Data variable length, depends on Data Type
- ……
- Nth Sub-Data Type – 1 byte
- Nth Sub-Data Length – 1 byte
- Nth Sub-Data– variable length, depends on Data Type N
- Error Detection Code – 1 byte

3.6.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII "M"
2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type (9)
6	Unit ID (total 32 bits)
7	
8	

9	
10	Command Numerator
11	Authentication Code
12	
13	
14	
15	Packet Control Field
16	Length (of the modules section - not including the checksum)
17	First Sub-data Type
18	First Sub-data Length
19	First Sub-data Data
…	…
	Nth Sub-data Type
	Nth Sub-data Length
	Nth Sub-data Data
Last Byte	Error Detection Code (8-bit additive checksum, excluding system code)

3.6.3 *Detailed Per-Field Specifications*

3.6.3.1 System Code

Refer to Section 3.2.3.1

3.6.3.2 Message Type

Modular message requests contain a value of 9 (nine) in the message type field.

3.6.3.3 Unit ID

Refer to Section 3.2.3.3

3.6.3.4 Command Numerator (Anti-Tango™)

Refer to Section 3.2.3.4

3.6.3.5 Authentication Code

Refer to Section 3.2.3.5

3.6.3.6 Packet Control Field

Direction	Out of space indication	Unused					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Direction

0 – Data from the unit

1 – Request (unit-bound)

Out of Space Indication

0 – All the requested data is present in the message.

1 – Some Sub-data was not returned due to data size.

3.6.3.7 Length

That field includes the number of data bytes of the modules (with their types and lengths). It is the number of bytes from byte 17 to the byte of the checksum, which is not included.

3.6.4 *Inbound Sub-Data Types Table*

Code (Hex)	Function
0x01	Firmware Platform Manifest Request
0x04	Time and Location Stamp Request
0x07	Usage Counter Request
0x08	Authentication Table Update Command
0x09	Cell ID Request
0x12	Modular Platform Manifest Request
0x14	Pulse Counter Measurement Request
0x19	One-Wire Temperature Sensors Measurement Request

3.6.5 *Firmware Platform Manifest Request*

This sub-data serves as a Firmware Manifest Request. The unit responds to this sub-data with Firmware Platform Manifest sub-data (0x01).

Byte	Description
0	Sub-Data Type (0x01)

Byte	Description
1	Length – 0

3.6.6 *Time and Location Stamp Request*

This sub-data serves as a Time and Location Stamp Request. The unit responds to this sub-data with Time and Location Stamp sub-data (0x04).

Byte	Description
0	Sub-Data Type (0x04)
1	Length – 0

3.6.7 *Usage Counter Request*

The purpose of this feature is to count the "high state" time of a pair of inputs, for example, to report the total engine hours of a machine.

The inputs whose "high state" time is counted are selectable by programming.

Two timers can be assigned to a specific input, including the option to assign both timers to the same input. Each input, including ignition, supports this "high state" time calculation.

The value of the measured time from each input is stored in RAM (protected, not erased on software reset, 24 bits for each parameter, not part of configuration memory), with a resolution of minutes.

The unit rounds off partial minutes: (1:29 is regarded as 1 minute and 1:30 and above as 2 minutes).

Once a day, the content of both usage counters is backed up on the dedicated address in non-volatile memory.

The timer proceeds with time counting (from the value stored in RAM) each time the logical level of the appropriate input changes from "low to high".

The timer stops counting each time the logical level of the input changes from "high" to "low".

The RAM values of usage counter are automatically updated on each RS232 and OTA “Counter’s Set” command.

Byte	Description
0	Sub-Data Type (0x07)
1	Length – 9

<table>
<tr><th>Byte</th><th colspan="8">Description</th></tr>
<tr><td rowspan="3">2</td><td colspan="8">Control Byte</td></tr>
<tr><td colspan="5">Spare</td><td>Enable Periodical Update
0 – Disable
1 – Enable</td><td colspan="2">Action Bits
0 – Read counters data
1 – Write counter 1
2 – Write counter 2
3 – Write counters 1 and 2</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>3</td><td colspan="8">Update Period (Minutes, 0 – cancels periodic reports)</td></tr>
<tr><td>4</td><td colspan="8">Spare</td></tr>
<tr><td>5</td><td colspan="8" rowspan="3">Counter 1 Value (Minutes)</td></tr>
<tr><td>6</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td><td colspan="8" rowspan="3">Counter 2 Value (Minutes)</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td></tr>
</table>

NOTE: If both Action bits are zero (request command) the Counter 1/2 Value fields are sent as zeros.

3.6.8 *Authentication Table Update Command*

The system provides protection against unapproved command attempts. For example, it provides protection against an attempt to change traffic destination IP by an unauthorized person. Every incoming message to the unit (such as command, acknowledge and so on) is provided a unique code, which is verified by the unit. If the code is not verified as authentic, the unit does not perform / acknowledge the command.

If Command Authentication is enabled in the unit's programming, the unit checks a valid 4-byte authentication code in bytes 11-14 of every inbound message. An inbound message with an invalid authentication code is declined by the unit. The unit does not respond to such a command and does not perform it. The 4 bytes authentication code in bytes 11-14 is generated as a function of two variables:

- Unit's ID
- 8 bytes Authentication Table, stored in the NVM of the unit and concurrently in the Communication Center application

----
NOTE: The OTA Authentication table modification will be accepted by the unit only if the Command Authentication feature is DISABLED in the unit's programming.

The default values of the Authentication Table (8 bytes, 16 nibbles) are as follows:

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Value	2	15	7	9	12	1	4	6	8	3	11	14	0	5	10	13

Note: The values in the authentication table must be different.

This sub-data is sent to the unit in order to access an Authentication Table values OTA (read, write or modify).

<table>
<tr><th>Byte</th><th colspan="8">Description</th></tr>
<tr><td>0</td><td colspan="8">Sub-Data Type (0x08)</td></tr>
<tr><td>1</td><td colspan="8">Length – 10</td></tr>
<tr><td rowspan="3">2</td><td colspan="8">Action Byte</td></tr>
<tr><td colspan="6">Spare</td><td colspan="2">Read/Write
0 - Read Authentication table from NVM
1 – Write Authentication table to NVM</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>3</td><td colspan="8">Spare</td></tr>
<tr><td>4</td><td colspan="4">Authentication table Index 0</td><td colspan="4">Authentication table Index 1</td></tr>
<tr><td>5</td><td colspan="4">Authentication table Index 2</td><td colspan="4">Authentication table Index 3</td></tr>
<tr><td>6</td><td colspan="4">Authentication table Index 4</td><td colspan="4">Authentication table Index 5</td></tr>
<tr><td>7</td><td colspan="4">Authentication table Index 6</td><td colspan="4">Authentication table Index 7</td></tr>
</table>

Byte	Description	
8	Authentication table Index 8	Authentication table Index 9
9	Authentication table Index 10	Authentication table Index 11
10	Authentication table Index 12	Authentication table Index 13
11	Authentication table Index 14	Authentication table Index 15

NOTE: Reset is required in order to apply OTA Authentication table modification.

3.6.9 *Cell ID Request*

This sub-data causes the unit to generate a Type-9 outbound message, containing the last known Cell ID related information (updated every 60 seconds). The generated message will vary according to unit Modem: 2G – sub-data 0x09, 3G – 0x0C, CDMA – 0x1C, 4G – 0x1D.

Byte	Description
0	Sub-Data Type (0x09)
1	Length – 2
2	Cell ID Request (0x09)
3	Spare

3.6.10 *Modular Platform Manifest Request*

This command causes the unit to generate an OTA Modular Platform Manifest message. The message will contain the data fields as per the specification in the command.

Data part: The data part of this packet has a size of 6 bytes. Each byte contains a bitmask as described below. Setting bit to "1" causes the unit to add a corresponding field to the Modular Platform Manifest.

Byte	Description
0	Sub-Data Type (0x12)
1	Length – 6

Byte	Description							
2	Bit Map 0							
	Size of external RAM	Amount of ext. non-volatile memory used by application (e.g. configuration)	Size of external non-volatile memory	Size of internal RAM	Amount of non-volatile memory used by application (e.g. configuration)	Size of Program memory	Accelerometer identifier	Processor identifier
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
3	Bit Map 1							
	FW Upgrade Date/Time	First Activation Date/Time	GPS firmware	GPS Type	Modem firmware	Modem type	Current Hardware ID number	Current Firmware ID number
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
4	Bit Map 2							
	Originating FW ID	IMSI/IMEI/MEID	Cello-CANiQ VIN (Vehicle ID)	DFD/SD card version	Boot loader ID	System ID (STM ID in case of STM controller)	Firmware name (string)	Last Configuration Change Date/Time
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
5	Bit Map 3							

Byte	Description							
	Maintenance Configuration	Modem Firmware Sub Version	PIN#8 HW Selected Function	SIM ICCID	BT Extender Identification	Size of internal non-volatile memory request	Version Information	Nano sensors
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6	Bit Map 4							
	Reserved	Reserved	Reserved	Reserved	CSR BT Module Identification	Battery Fuel Gauge Identification	Silicon Labs 4632 receiver version	BT version number
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7	Bit Map 5							
	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Note: Reserved bits will be sent as 0.

3.6.11 *Pulse Counter Measurement Request*

Pulse counter measurement request is mainly used for fuel consumption measurement. The measurement is taken from the last time the counter was reset.

Byte	Description
0	Sub-Data Type (0x14)
1	Length – 2
2	Spare
3	Spare

3.6.12 *One-Wire Temperature Sensor Measurement Request*

This message is sent by the server towards the unit for requesting One-Wire temperature sensor measurement results.

Byte	Description
0	Sub-Data Type (0x19)
1	Length – 2
2	Spare
3	Spare

3.7 Modular Message Request (Message Type 11)

Type 11 was introduced for supporting true modular protocol. The basic structure of the protocol is designed to carry records with predefined structure called modules. The protocol will be used as an extension for Cello fleet protocol. Type 11 supports theoretical message length of up to 65536 bytes, tough the actual rate will be constrained by the HW limitations.

3.7.1 *Message Ingredients*

- Message header
  - System Code – 4 bytes
  - Message Type – 1 byte
  - Destination Unit ID – 4 bytes
  - Command Numerator – 1 byte
- Authentication Code – 4 bytes
- Packet Control Field – Legacy fleet field
- Message length – 2 bytes
- spare – 4 bytes
- Payload Modules – User Configuration Depended
- Error Detection Code – 1 byte

3.7.2 *Byte-Aligned Table*

Byte	Description
1	System Code, byte 1 – ASCII “M”

2	System Code, byte 2 – ASCII "C"
3	System Code, byte 3 – ASCII "G"
4	System Code, byte 4 – ASCII "P"
5	Message Type (11)
6	Unit ID
7	
8	
9	
10	Command Numerator (When transmitting ACK/NACK packet, it carries the numerator of the original message)
11	Authentication Code
12	
13	
14	
15	Packet Control Field
16	Length (of the modules section - not including the checksum). Must be ≠ 0 (to symbolize inbound message), meaning that there should not be a message without any modules.
17	
18	Spare (sent as 0)
19	
20	
21	
22	Modules
…	…
…	…
Last Byte	Error Detection Code (8-bit additive checksum, excluding system code)

3.7.3 *Detailed Per-Field Specifications*

3.7.3.1 System Code

Refer to Section 3.2.3.1

3.7.3.2 Message Type

Modular message requests contain a value of 11 (eleven) in the message type field.

3.7.3.3 Unit ID

Refer to Section 3.2.3.3

3.7.3.4 Command Numerator (Anti-Tango™)

Refer to Section 3.2.3.4

3.7.3.5 Authentication Code

Refer to Section 3.2.3.5

3.7.3.6 Packet Control Field

Direction	Out of space indication	Unused					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Direction

0 – Data from the unit

1 – Request (unit-bound)

Out of Space Indication

0 – All the requested data is present in the message.

1 – Some Sub-data was not returned due to data size.

3.7.3.7 Length

That field includes the number of data bytes of the modules (with their codes and lengths). It is the number of bytes from byte 18 to the byte of the checksum, which is not included.

3.7.4 *Inbound Type 11 Module Structure*

The general structure of a type 11 module is as follows:

Byte	Description
0	Module Code
1	Length of module (16 bits) – Number of payload bytes
2	
3	Module Payload bytes
.	

.	
n	

3.7.5 *Inbound Type 11 Modules Table*

Code	Description
9	ACK (from server)
10	Configuration Memory Write
11	Configuration Memory Block Request
13	Authenticated Features Command
16	Modem FOTA Command
29	General Module Query
32	General Command

3.7.6 *ACK/NACK*

Byte	Description
0	Module 9 – ACK/NACK
1	Length of module – 3
2	
3	0 - ACK
4	Spare
5	Spare

3.7.7 *Configuration Memory Write*

Byte	Description
0	Module 10 - Configuration Memory Write
1	Length of module – Variable
2	
3	Numerator
4	
5	Number of instances

6	Memory type – 0	
7	Memory entry unit type 0 – Bit 1 – Byte 2 – Word (16 bits) 3 – Double Word (32 bits) (Only Byte entry unit type is currently supported)	Instance 1
8	Address in the configuration memory space	
9		
10		
11		
12	Number of Entries	
13		
…	Data payload (according to the entry size and the number of entries defined above)	
…	…	Instance 2
…	…	
…	…	

3.7.8 *Configuration Memory Read Request*

Byte	Description	
0	Module 11 - Configuration Memory Read Request	
1	Length of module – Variable	
2		
3	Numerator	
4		
5	Number of instances	
6	Memory type – 0	
7	Memory entry unit type 0 – Bit	Instance 1

	1 – Byte 2 – Word (16 bits) 3 – Double Word (32 bits) (Only Byte entry unit type is currently supported)	
8	Address in the configuration memory space	
9		
10		
11		
12	Number of Entries	
13		
…	…	Instance 2
…	…	
…	…	

3.7.9 *Authenticated Features Command*

This module enables query/activation/de-activation of features in the unit. It contains the desired features codes.

On query command, there will be no feature codes.

On activation/de-activation command the unit will reply with ACK/NACK (module 9), while on query command the unit will reply with Authenticated Features Query Response (module 13).

Byte	Description
0	Module 13 – Authenticated Features Command
1	Length of module – Variable
2	
3	Control Byte <table><tr><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td colspan="2">Command Code 0 - Query 1 - Activation 2 - De-Activation</td></tr><tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr></table>
4	Spare

5	Number of Feature Codes (0 for query command, 1-8 for activation/de-activation)	
6	C[0]	Feature Code 1
7	C[1]	
8	C[2]	
9	C[3]	
10	Spare	
11	Spare	
12	Spare	
13	Spare	
14	Spare	
15	Spare	
16	C[0]	Feature Code 2
17	C[1]	
18	C[2]	
19	C[3]	
20	Spare	
21	Spare	
22	Spare	
23	Spare	
24	Spare	
25	Spare	
…	…	
…	C[0]	Feature Code n
	C[1]	
	C[2]	
	C[3]	
	Spare	
	Spare	

	Spare	
	Spare	
	Spare	
	Spare	

3.7.10 *Modem FOTA Command*

This module enables activation of Modem FOTA upgrade via FTP server (which hosts a delta file received from advance). It contains the details of the desired FTP server, directory, and upgrade file.

Upon receiving this command, the unit will send ACK (module 9).

After completing the upgrade process, the unit will reply on this command with Modem FOTA Response (module 16).

<table>
<tr><th>Byte</th><th colspan="8">Description</th></tr>
<tr><td>0</td><td colspan="8">Module 16 – Modem FOTA Command</td></tr>
<tr><td>1</td><td colspan="8" rowspan="2">Length of module – 195</td></tr>
<tr><td>2</td></tr>
<tr><td rowspan="3">3</td><td colspan="8">Control Byte</td></tr>
<tr><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td><td>Spare</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>4</td><td colspan="8">Spare</td></tr>
<tr><td>5</td><td colspan="8">Spare</td></tr>
<tr><td>6</td><td colspan="8">FTP Server IP Address Byte 0 (LSB)</td></tr>
<tr><td>7</td><td colspan="8">FTP Server IP Address Byte 1</td></tr>
<tr><td>8</td><td colspan="8">FTP Server IP Address Byte 2</td></tr>
<tr><td>9</td><td colspan="8">FTP Server IP Address Byte 3 (MSB)</td></tr>
<tr><td>10</td><td colspan="8" rowspan="2">FTP Server IP Port (0-65535)</td></tr>
<tr><td>11</td></tr>
<tr><td>12</td><td colspan="8">Spare</td></tr>
<tr><td>13</td><td colspan="8">Spare</td></tr>
</table>

14-43	FTP Server Username (ASCII String, length 30, zero padded)
44-73	FTP Server Password (ASCII String, length 30, zero padded)
74	Spare
75	Spare
76-145	Full Path (ASCII String, length 70, zero padded)
146	Spare
147	Spare
148-197	Full File Name (ASCII String, length 50, zero padded)

3.7.11 *General Module Query*

This command will be sent by the server to request a set of outbound modules to be returned to the server. The module describes a list of module IDs. The addressed unit will respond with a type 11 message carrying the requested modules content arranged in the same order of the request.

Byte	Description
0	Module 29 – General Module Query
1	Length of module – Variable
2	
3	Number of requested Modules
4	First requested module ID
5	Second requested module ID
…	…

3.7.12 *General Command*

This module enables the server to command the unit to perform multiple actions while specifying the action code and optional data bytes attached to the command. The general format of the module is shown below. The unit will send Acknowledge (outbound type 11 module 9) upon receiving this module.

Byte	Description
0	Module 32 – General Command
1	Length of module – Variable
2	

3	Number of Command entries	
4	Command ID	Command Entry 1
5		
…	Command data bytes	
…	Command ID	Command Entry 2
…		
…	Command data bytes	
…	…	

Commands Types Description:

Command ID	Description	Number of Attached data bytes	Expected unit response
1	Reset Unit	0	One ACK for all the command request