P

u-blox ZOE-M8B

Receiver description

Including protocol specification

Abstract

The receiver description including protocol specification describes the firmware features, specifications and configuration for u-blox ZOE-M8B high performance positioning SiP module.
The receiver description provides an overview and conceptual details of the supported features.
The protocol specification describes the NMEA and RTCM protocols as well as the UBX protocol (version 23.01) and serves as a reference manual.
It includes the standard precision GNSS products with the Super-Efficient (Super-E) power saving features.

www.u-blox.com

UBX-18031559 - R01

Document Information		
Title	**u-blox ZOE-M8B Receiver description**	
Subtitle	Including protocol specification v23.01	
Document type	Interface manual	
Document number	UBX-18031559	
Revision and date	R01 (5c54fb0M)	30 August 2019
Document status	Production information	

Document status explanation	
Objective Specification	Document contains target values. Revised and supplementary data will be published later.
Advance Information	Document contains data based on early testing. Revised and supplementary data will be published later.
Early Production Information	Document contains data from product verification. Revised and supplementary data may be published later.
Production Information	Document contains the final product specification.

This document applies to the following products:

Product name	Type number	Firmware version	Product category
ZOE-M8B	ZOE-M8B-0-10	SPG 3.51	Standard Precision GNSS

Table of Contents

Preface

1 Document Overview

The Interface Description Including Receiver Description is an important resource for integrating and configuring u-blox receivers. This document has a modular structure and it is not necessary to read it from the beginning to the end. There are two main sections: The Receiver Description and the Interface Description.

The Receiver Description describes the software aspects of system features and configuration of u-blox receivers. The Receiver Description is structured according to areas of functionality, with links provided to the corresponding NMEA and UBX messages, which are described in the Interface Description.

The Interface Description is a reference describing the messages used by the u-blox receiver and is organized by the specific NMEA, UBX, and RTCM messages.

2 Firmware and Protocol Versions

The protocol version defines a set of messages that are applicable across various u-blox products. Each firmware used by a u-blox receiver supports a specific protocol version, which is not configurable.

The following sections will explain how to decode the shown information to get the firmware and the protocol version.

2.1 How to Determine the Version and the Location of the Firmware

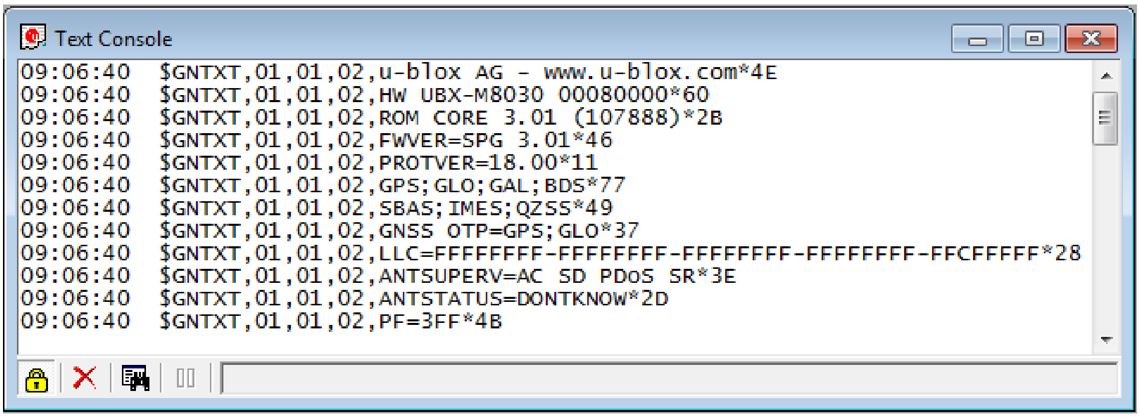
The u-blox receiver can run a firmware from two different locations:

- Internal ROM
- External Flash memory

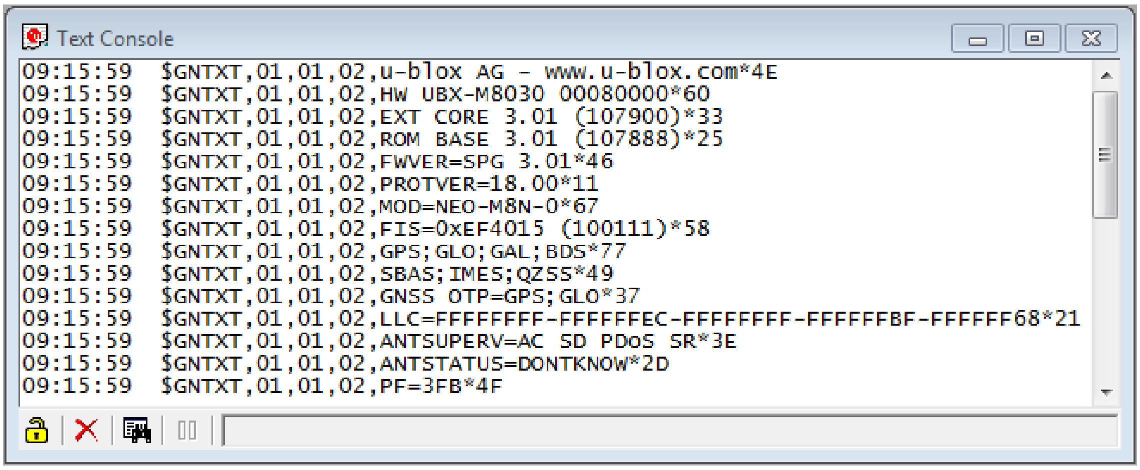
The location and the version of the currently running firmware can be found in the boot screen or in the UBX-MON-VER message.

2.1.1 Decoding the Boot Screen (for Protocol Version 18 and Above)

Boot screen for a u-blox receiver running from ROM:



Boot screen for a u-blox receiver running from Flash:



Not every line is output by every u-blox receiver in the boot screen. This depends on the product, the firmware location and the firmware version.

Possible lines in the boot screen and their meanings:

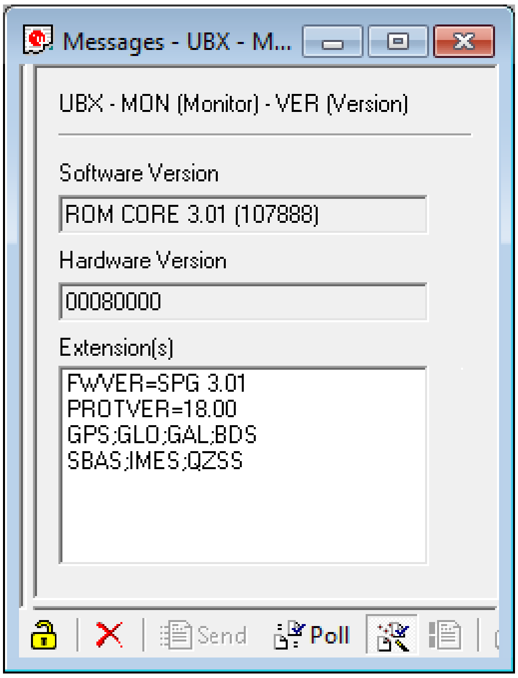
Entry	Description
`u-blox AG - www.u-blox.com`	Start of the boot screen
`HW UBX-M8030 00800000`	Hardware version of the u-blox receiver (u-blox M8 receiver)
`HW UBX-G8020 00800000`	Hardware version of the u-blox receiver (u-blox 8 receiver)
`ROM CORE 3.01 (107888)`	Firmware version 3.01 running from **ROM** (revision number)
`EXT CORE 3.01 (107900)`	Firmware version 3.01 running from **Flash** (revision number)
`ROM BASE 3.01 (107888)`	Underlying firmware version 3.01 in **ROM** (revision number)
`FWVER=SPG 3.01`	Firmware of product category and version where SPG: Firmware of Standard Precision GNSS product HPG: Firmware of High Precision GNSS product ADR: Firmware of ADR product UDR: Firmware of UDR product TIM: Firmware of Time Sync product FTS: Firmware of Time & Frequency Sync product
`PROTVER=18.00`	Supported protocol version
`MOD=NEO-M8N-0`	Module identification. Set in production.
`FIS=0xEF4015 (100111)`	Flash Information Structure (FIS) file for Flash memory with JEDEC `0xEF4015` found in the external flash memory. Revision number of the file is indicated in brackets.
`GPS;GLO;GAL;BDS`	Supported Major GNSS.
`SBAS;IMES;QZSS`	Supported Augmentation systems.
`GNSS OTP=GPS;GLO`	Default Major GNSS selection.
`LLC FFFFFFFF-FFFFFFFF-FFFFFFFF-FFFFFFFF-FFCFFFFF`	Low-level configuration of the u-blox receiver.
`ANTSUPERV=AC SD PDoS SR`	Configuration of the Antenna supervisor where AC: Active Antenna Control enabled SD: Short Circuit Detection enabled OD: Open Circuit Detection enabled PDoS: Short Circuit Power Down Logic enabled SR: Automatic Recovery from Short state
`PF=3FF`	Product configuration.

☞ The line containing the **FWVER** indicates which version of the firmware is currently running and is called **firmware version** in the rest of the document.

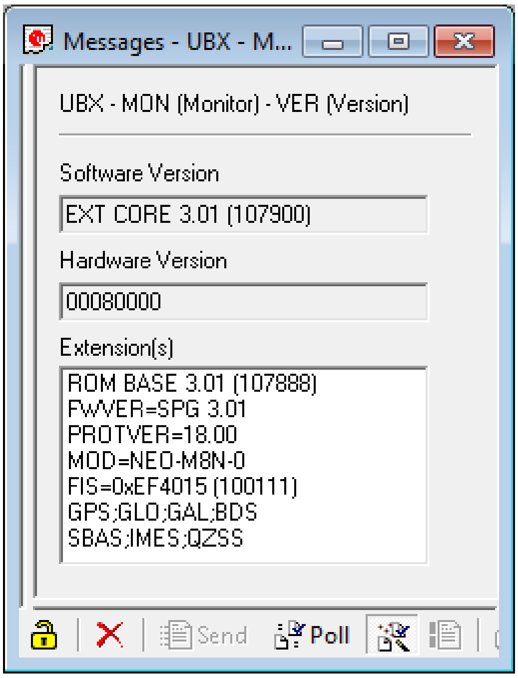
☞ The numbers in parentheses (revision numbers) should only be used to identify a known firmware version and are not guaranteed to increase over time.

2.1.2 Decoding the output of UBX-MON-VER (for Protocol Version 18 and above)

UBX-MON-VER for receiver running from ROM



UBX-MON-VER for receiver running from Flash



Possible fields in UBX-MON-VER and their meanings:

Entry	Description

Software Version
`ROM CORE 3.01 (107888)`
`EXT CORE 3.01 (107900)`
	Currently running firmware version.
	If `ROM CORE`, then the u-blox receiver runs from **ROM**.
	If `EXT CORE`, then the u-blox receiver runs from **Flash**.

Hardware Version	The hardware version of the u-blox receiver.
Extension(s)	Extended information about the u-blox receiver firmware. See table below for the entries.

☞ Not every entry is output by every u-blox receiver in the UBX-MON-VER extensions. This depends on the product, the firmware location and the firmware version.

Possible entries in UBX-MON-VER Extension(s):

Entry	Description
`ROM BASE 3.01 (107888)`	Underlying firmware version in ROM. If such an entry is present, then the u-blox receiver runs from **Flash**.

Possible entries in UBX-MON-VER Extension(s): continued

Entry	Description
FWVER=SPG 3.01	Firmware of product category and version where SPG: Firmware of Standard Precision GNSS product HPG: Firmware of High Precision GNSS product ADR: Firmware of ADR product UDR: Firmware of UDR product TIM: Firmware of Time Sync product FTS: Firmware of Time & Frequency Sync product
PROTVER=18.00	Supported protocol version.
MOD=NEO-M8N-0	Module identification. Set in production.
FIS=0xEF4015 (100111)	Flash Information Structure (FIS) file for Flash memory with JEDEC 0xEF4015 found in the external flash memory. Revision number of the file is indicated in brackets.
GPS;GLO;GAL;BDS	Supported Major GNSS.
SBAS;IMES;QZSS	Supported Augmentation systems.

2.2 How to Determine the Supported Protocol Version of the u-blox Receiver

Each u-blox receiver reports its supported protocol version in the following ways:

- On start-up in the boot screen
- In the UBX-MON-VER message

with the line containing PROTVER (example: PROTVER=18.00).

Additionally, the firmware string, together with the firmware version, can be used to look up the corresponding protocol version. The tables below give an overview of the released firmware and their corresponding protocol versions.

2.2.1 u-blox 8 / u-blox M8 Firmware and Supported Protocol Versions

Firmware for Standard Precision GNSS products

Firmware version	Firmware string	Protocol Version
SPG 2.01	ROM CORE 2.01 (75331) Oct 29 2013 13:28:17	15.00
SPG 2.01	EXT CORE 2.01 (75350) Oct 29 2013 16:15:41	15.00
SPG 3.01	ROM CORE 3.01 (107888)	18.00
SPG 3.01	EXT CORE 3.01 (107900)	18.00
SPG 3.50	EXT CORE 3.50 (190461)	23.00
SPG 3.51	ROM CORE 3.51 (19dc23)	23.01
SPG 3.51	EXT CORE 3.51 (19dc23)	23.01

Firmware for High Precision GNSS Products

Firmware version	Firmware string	Protocol Version
HPG 1.00	EXT CORE 3.01 (111160)	20.00
HPG 1.11	EXT CORE 3.01 (b8bc67)	20.01
HPG 1.20	EXT CORE 3.01 (d34ed4)	20.10
HPG 1.30	EXT CORE 3.01 (d080e3)	20.20
HPG 1.40	EXT CORE 3.01 (db0c89)	20.30

Firmware for Dead Reckoning products

Firmware version	Firmware string	Protocol Version
ADR 3.00	EXT CORE 2.01 (77076) Dec 18 2013 09:40:24 ADR 3.00	15.00
ADR 3.10	EXT CORE 2.01 (87683) Nov 21 2014 14:03:10 ADR 3.10 M8L	15.01
ADR 3.11	EXT CORE 2.01 (89981) Jan 20 2015 17:22:06 ADR 3.11 M8L	15.01
ADR 4.00	EXT CORE 3.01 (16559bf) Apr 21 2016 15:49:07 ADR 4.00	19.00
ADR 4.10	EXT CORE 3.01 (c0c787c) Apr 24 2017 17:31:42 ADR 4.10	19.10
ADR 4.11	EXT CORE 3.01 (d189ff) Aug 22 2017 14:40:05 ADR 4.11	19.10
UDR 1.00	EXT CORE 3.01 (16559bf) Apr 21 2016 15:50:59 UDR 1.00	19.00

Firmware for Timing products

Firmware version	Firmware string	Protocol Version
FTS 1.01	EXT CORE 2.20 (81289) May 14 2014 14:11:24	16.00
TIM 1.00	EXT CORE 2.30 (85522) Sep 29 2014 09:40:12	17.00
TIM 1.01	EXT CORE 2.30 (86283) Oct 20 2014 13:51:49	17.00
TIM 1.02	EXT CORE 2.30 (93796) Apr 8 2015 15:53:38	17.00
TIM 1.10	EXT CORE 3.01 (111141)	22.00

Receiver Description

3 Receiver Configuration

3.1 Configuration Concept

u-blox receivers are fully configurable with UBX protocol configuration messages (message class UBX-CFG). The configuration used by the u-blox receiver during normal operation is termed "Current Configuration". The Current Configuration can be changed during normal operation by sending any UBX-CFG-XXX message to the u-blox receiver over an I/O port. The u-blox receiver will change its Current Configuration immediately after receiving the configuration message. The u-blox receiver always uses only the Current Configuration.

Unless the Current Configuration is made permanent by using `UBX-CFG-CFG` as described below, the Current Configuration will be lost when there is:

- a power cycle
- a hardware reset
- a (complete) controlled software reset

See the section on resetting a u-blox receiver for details.

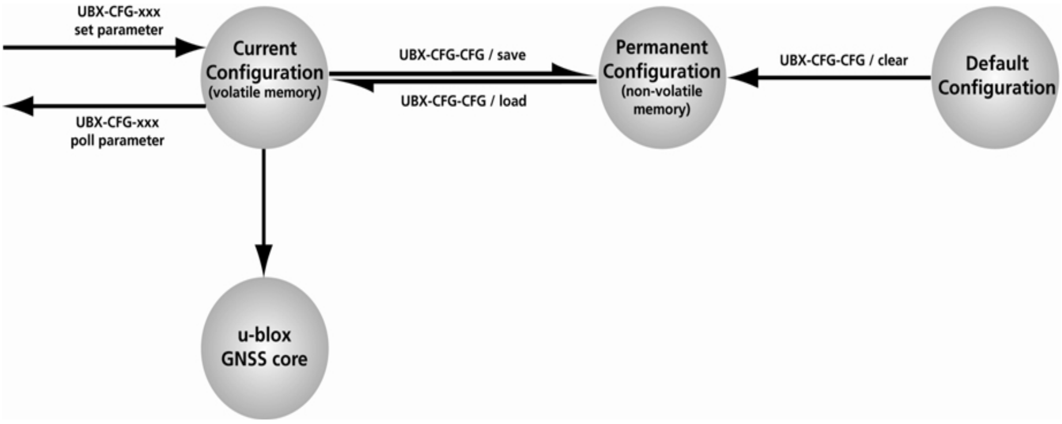
The Current Configuration can be made permanent (stored in a non-volatile memory) by saving it to the "Permanent Configuration". This is done by sending a `UBX-CFG-CFG` message with an appropriate **saveMask** (UBX-CFG-CFG/save).

The Permanent Configuration is copied to the Current Configuration during start-up or when a `UBX-CFG-CFG` message with an appropriate **loadMask** (UBX-CFG-CFG/load) is sent to the u-blox receiver.

The Permanent Configuration can be restored to the u-blox receiver's Default Configuration by sending a `UBX-CFG-CFG` message with an appropriate **clearMask** (UBX-CFG-CFG/clear) to the u-blox receiver. This only replaces the Permanent Configuration, not the Current Configuration. To make the u-blox receiver operate with the Default Configuration which was restored to the Permanent Configuration, a UBX-CFG-CFG/load command must be sent or the u-blox receiver must be reset.

The mentioned masks (saveMask, loadMask, clearMask) are 4-byte bitfields. Every bit represents one configuration sub-section. These sub-sections are defined in section "Organization of the Configuration Sections". All three masks are part of every UBX-CFG-CFG message. Save, load and clear commands can be combined in the same message. Order of execution is: clear, save, load.

The following diagram illustrates the process:



It is possible to change the current communications port settings using a UBX-CFG-CFG message. This could affect baud rate and other transmission parameters. Because there may be messages queued for transmission there may be uncertainty about which protocol applies to such messages. In addition a message currently in transmission may be corrupted by a protocol change. Host data reception parameters may have to be changed to be able to receive future messages, including the acknowledge message associated with the UBX-CFG-CFG message.

3.2 Organization of the Configuration Sections

The configuration is divided into several sub-sections. Each of these sub-sections corresponds to one or several UBX-CFG-XXX messages. The sub-section numbers in the following tables correspond to the bit position in the masks mentioned above. All values not listed are reserved

Configuration sub-sections

Number	Name	CFG messages	Description
0	PRT	UBX-CFG-PRT UBX-CFG-USB	Port and USB settings
1	MSG	UBX-CFG-MSG	Message settings (enable/disable, update rate)
2	INF	UBX-CFG-INF	Information output settings (Errors, Warnings, Notice, Test etc.)
3	NAV	UBX-CFG-NAV5 UBX-CFG-NAVX5 UBX-CFG-DAT UBX-CFG-RATE UBX-CFG-SBAS UBX-CFG-NMEA UBX-CFG-TMODE2	Settings for Navigation Parameters, Receiver Datum, Measurement and Navigation Rate, SBAS, NMEA protocol and Time mode (Timing products only)
4	RXM	UBX-CFG-GNSS UBX-CFG-TP5 UBX-CFG-RXM UBX-CFG-PM2 UBX-CFG-ITFM	GNSS Settings, Power Mode Settings, Time Pulse Settings, Jamming/Interference Monitor Settings
9	RINV	UBX-CFG-RINV	Remote Inventory configuration
10	ANT	UBX-CFG-ANT	Antenna configuration
11	LOG	UBX-CFG-LOGFILTER	Logging configuration

Configuration sub-sections continued

Number	Name	CFG messages	Description
12	FTS	UBX-CFG-DOSC UBX-CFG-ESRC UBX-CFG-SMGR	Disciplining configuration. Only applicable to the Time & Frequency Sync product.

3.3 Permanent Configuration Storage Media

The Current Configuration is stored in the volatile RAM of the u-blox receiver. Hence, any changes made to the Current Configuration without saving will be lost if any of the reset events listed in the section above occur. By using UBX-CFG-CFG/save, the selected configuration sub-sections are saved to all non-volatile memories available:

- On-chip BBR (battery backed RAM). In order for the BBR to work, a backup battery must be applied to the u-blox receiver.
- External flash memory, where available.

3.4 u-blox Receiver Default Configuration

The Permanent Configuration can be reset to Default Configuration through a UBX-CFG-CFG/clear message. The Default Configuration of the u-blox receiver is normally determined when the u-blox receiver is manufactured. Refer to specific product data sheet for further details.

3.5 Save-on-Shutdown Feature

The save-on-shutdown feature (SOS) enables the u-blox receiver to store the contents of the battery-backed RAM to external flash memory and restore it upon startup. This allows the u-blox receiver to preserve some of the features available only with a battery backup (preserving configuration and satellite orbit knowledge) without having a battery backup supply present. It does not, however, preserve any kind of time knowledge. The save-on-shutdown must be commanded by the host. The restore-on-startup is automatically done if the corresponding data is present in the flash. No expiration check of the data is done.

The following outlines the suggested shutdown procedure when using the save-on-shutdown feature:

- With the UBX-CFG-RST message, the host commands the u-blox receiver to stop, specifying reset mode 0x08 ("Controlled GNSS stop") and a BBR mask of 0 ("Hotstart").
- The u-blox receiver confirms the reception of a valid / invalid request with a UBX-ACK-ACK / UBX-ACK-NAK message.
- The host commands the saving of the contents of BBR to the flash memory using the UBX-UPD-SOS-BACKUP message.
- The u-blox receiver confirms the reception of a valid / invalid request with a UBX-ACK-ACK / UBX-ACK-NAK message.
- For a valid request the u-blox receiver reports on the success of the backup operation with a UBX-UPD-SOS-ACK message.
- The host powers off the u-blox receiver.

And consequently the startup procedure is as follows:

- The host powers on the u-blox receiver.
- The u-blox receiver detects the previously stored data in flash. It restores the corresponding

memory and reports the success of the operation with a UBX-UPD-SOS-RESTORED message on the port it had received the save command message (if the output protocol filter on that port allows it). It does not report anything if no stored data has been detected.

- Additionally the u-blox receiver outputs a UBX-INF-NOTICE and/or a NMEA-TXT message with the contents RESTORED in the boot screen (depends on port and information messages configuration) upon success.
- Optionally the host can deliver coarse time assistance using UBX-MGA-INI-TIME_UTC for better startup performance.

Once the u-blox receiver has started up it is suggested to delete the stored data using a UBX-UPD-SOS-CLEAR message. The u-blox receiver responds with a UBX-ACK-ACK or UBX-ACK-NAK message.

⚠ Note that this feature must not be used with Power Save Mode and that saved data must be deleted before switching to that mode.

4 Concurrent GNSS

Many u-blox positioning modules and chips are multi-GNSS receivers capable of receiving and processing signals from multiple Global Navigation Satellite Systems (GNSS).

u-blox concurrent GNSS receivers are multi-GNSS receivers that can acquire and track satellites from more than one GNSS system at the same time, and utilize them in positioning.

4.1 GNSS Types

u-blox receivers support a wide range of different GNSS. Some GNSS have large numbers of satellites deployed globally and therefore are generally capable of providing navigation solutions on their own. u-blox designates these as "major GNSS". By contrast, some are designed to be used to enhance the use of one or more major GNSS and u-blox designates these "augmentation systems".

In many cases, such as Satellite Numbering, this distinction does not matter as u-blox receivers generally try to combine information from all available GNSS to create the best possible navigation information. However, particularly in relation to configuring the receiver, the distinction can be important.

4.1.1 Major GNSS

The major GNSS supported by u-blox receivers are described below.

4.1.1.1 GPS

The Global Positioning System (GPS) is a GNSS operated by the US department of defense. Its purpose is to provide position, velocity and time for civilian and defense users on a global basis. The system currently consists of 32 medium earth orbit satellites and several ground control stations.

4.1.1.2 GLONASS

GLONASS is a GNSS operated by Russian Federation department of defense. Its purpose is to provide position, velocity and time for civilian and defense users on a global basis. The system consists of 24 medium earth orbit satellites and ground control stations.

It has a number of significant differences when compared to GPS. In most cases, u-blox receivers operate in a very similar manner when they are configured to use GLONASS signals instead of

GPS. However some aspects of receiver output are likely to be noticeably affected.

4.1.1.3 Galileo

> ⚠ At the time of writing (early 2016), the Galileo system was still under development with only a few fully operational SVs. Therefore, the precise performance and reliability of u-blox receivers when receiving Galileo signals is effectively impossible to guarantee.

Galileo is a GNSS operated by the European Union. Its purpose is to provide position, velocity and time for civilian users on a global basis. The system is currently not fully operational. It is eventually expected to consist of 30 medium earth orbit satellites.

On u-blox M8 receivers a maximum of ten channels can be assigned to Galileo for signal acquisition and tracking. Note that at most eight Galileo satellites will be used for navigation. It is recommended not to set the number of Galileo channels higher than eight in `UBX-CFG-GNSS`.

4.1.1.3.1 Search and Rescue Return Link Message

The receiver supports reception and output of Search and Rescue (SAR) Return Link Messages (RLM). When enabled, a `UBX-RXM-RLM` message will be generated whenever an RLM is detected by the receiver.

> ⚠ At the time of writing (early 2016), no live transmission of RLMs by Galileo SVs had been observed, so the details of their use was impossible to verify completely.

4.1.1.4 BeiDou

BeiDou is a GNSS operated by China. Its purpose is to initially provide position, velocity and time for users in Asia. In a later stage when the system is fully deployed it will have worldwide coverage. The full system will consist of five geostationary, five inclined geosynchronous and 27 medium earth orbit satellites, as well as control, upload and monitoring stations. Although this implies a full constellation of 37 SVs, only SVs numbered 1 to 30 are fully supported in the D1/D2 NAV message described by the Interface Control Document version 2.0. For SVs numbered above 30, there is currently no almanac or differential correction. Consequently, u-blox receivers only use BeiDou SVs numbered 1 to 30.

4.1.2 Augmentation Systems

The augmentation systems supported by u-blox receivers are described below.

4.1.2.1 SBAS

There are a number of Space Based Augmentation Systems (SBAS) operated by different countries using geostationary satellites. u-blox receivers currently support the following:

- WAAS (Wide Area Augmentation System) operated by the US.
- EGNOS (European Geostationary Navigation Overlay Service) operated by the EU.
- MSAS (Multi-functional Satellite Augmentation System) operated by Japan.
- GAGAN (GPS Aided Geo Augmented Navigation) operated by India.

See section SBAS for more details.

4.1.2.2 QZSS

The Quasi Zenith Satellite System (QZSS) is a regional satellite augmentation system operated by Japan Aerospace Exploration Agency (JAXA). It is intended as an enhancement to GPS, to increase availability and positional accuracy. The QZSS system achieves this by transmitting

GPS-compatible signals in the GPS bands.

NMEA messages will show the QZSS satellites only if configured to do so (see section Satellite Numbering).

The QZSS L1SAIF is an additional signal broadcast by QZSS satellites that contains augmentation and other data.

4.1.2.3 IMES

The Indoor MEssaging System (IMES) is an extension to the QZSS specification.

See section IMES for more details.

4.2 Configuration

The `UBX-CFG-GNSS` message allows the user to specify which GNSS signals should be processed along with limits on how many tracking channels should be allocated to each GNSS. The receiver will respond to such a request with a `UBX-ACK-ACK` message if it can support the requested configuration or a `UBX-ACK-NAK` message if not.

☞ Customers enabling BeiDou and/or Galileo who wish to use the NMEA protocol are recommended to select NMEA version 4.1, as earlier versions have no support for these two GNSS. See the NMEA protocol section for details on selecting NMEA versions.

The combinations of systems which can be configured simultaneously depends on the receivers capability to receive several carrier frequencies. The `UBX-MON-GNSS` message reports which major GNSS can be selected. Please refer to the data sheet of the corresponding u-blox receiver for full information. Usually GPS, SBAS (e.g. WAAS, EGNOS, MSAS), QZSS and Galileo can be enabled together, because they all use the 1575.42MHz L1 frequency. GLONASS and BeiDou both operate on different frequencies, therefore the receiver must be able to receive a second or even third carrier frequency in order to process these systems together with GPS.

☞ It is recommended to disable GLONASS and BeiDou if a GPS-only antenna or GPS-only SAW filter is used.

In all circumstances, it is necessary for at least one major GNSS to be enabled. It is also required that at least 4 tracking channels are available to each enabled major GNSS, i.e. `maxTrkCh` must have a minimum value of 4 for each enabled major GNSS. Further requirements on generating configurations acceptable by the receiver can be found in `UBX-CFG-GNSS`.

4.2.1 Switching between GNSS

Users should be aware that switching between GNSS (and especially away from GPS) may affect the long term accuracy of the receiver until the next cold start. In normal operation the receiver selects the best models and corrections from the transmitted auxiliary data (e.g. UTC and Ionospheric parameters), basing this selection on the configured GNSS. Disabling a major GNSS prevents auxiliary data from that GNSS being refreshed and so it will become stale, resulting in progressively degraded performance. This can occur even if the main power supply is removed, as most receivers retain auxiliary data in non-volatile storage, e.g. Battery Backed RAM (BBR). For this reason, u-blox recommends that receivers are cold started after any change that disables an active GNSS, within a few weeks, but preferably immediately. This will ensure that the receiver then uses only regularly refreshed information from the newly configured constellations.

4.2.2 Configuring QZSS L1SAIF

By default the receiver will be configured for QZSS L1C/A, this can be changed so the receiver can be configured for QZSS L1SAIF also. See the table below for `UBX-CFG-GNSS` `sigCfgMask` settings for signals on QZSS. For example, to enable QZSS L1C/A and QZSS L1SAIF, set the `gnssId` to 5 (for QZSS) and `sigCfgMask` to 0x05. If supported by the firmware, L1SAIF would then be enabled.

QZSS Signal configuration for UBX-CFG-GNSS

GnssId	Description	Signal mask
5	QZSS	0x01 = QZSS L1C/A 0x04 = QZSS L1SAIF

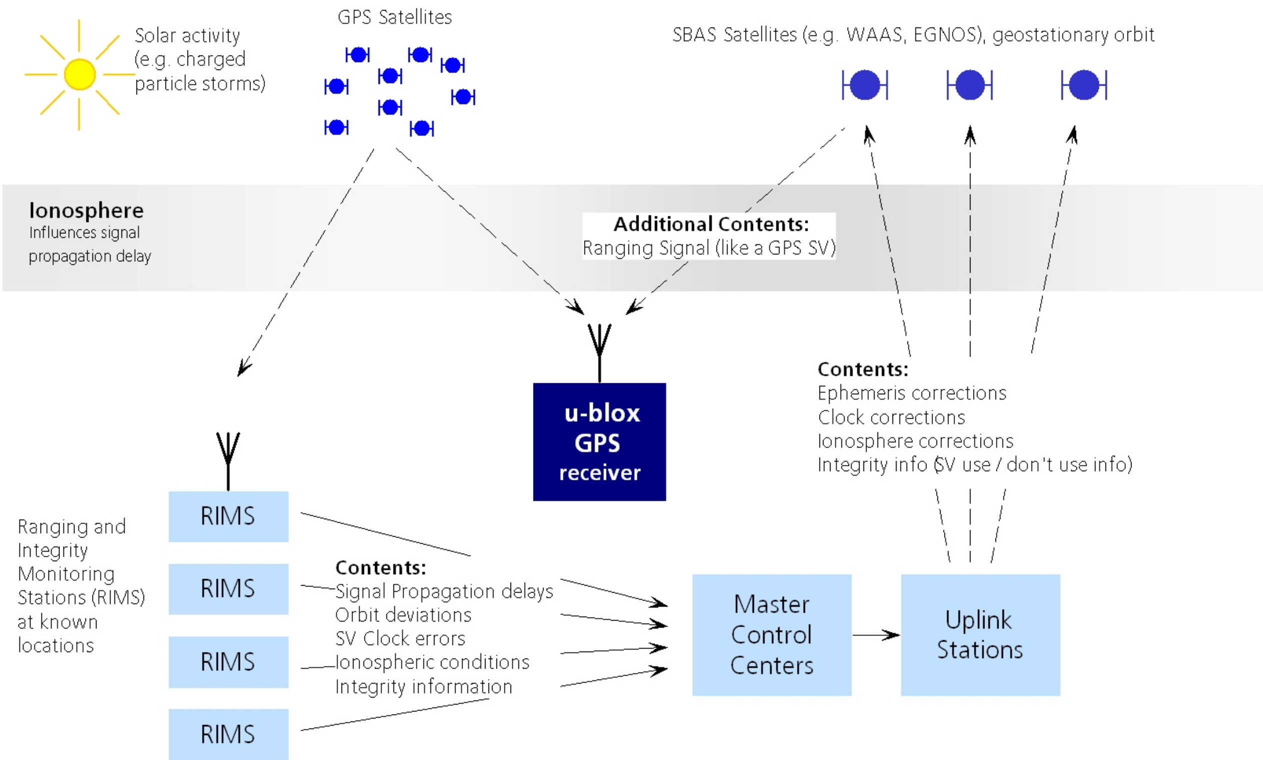
5 SBAS Configuration Settings Description

5.1 SBAS (Satellite Based Augmentation Systems)

SBAS (Satellite Based Augmentation System) is an augmentation technology for GPS, which calculates GPS integrity and correction data with RIMS (Ranging and Integrity Monitoring Stations) on the ground and uses geostationary satellites to broadcast GPS integrity and correction data to GPS users. The correction data is transmitted on the GPS L1 frequency (1575.42 MHz), and therefore no additional receiver is required to make use of the correction and integrity data.

☞ u-blox receivers will only process corrections for GPS. Other corrections are not applied, even if, as planned, some SBAS satellites start to transmit them (e.g. SDCM for GLONASS).

SBAS Principle



There are several compatible SBAS systems available or in development all around the world:

- WAAS (Wide Area Augmentation System) for North America has been in operation since 2003.
- MSAS (Multi-Functional Satellite Augmentation System) for Japan has been in operation since 2007.
- EGNOS (European Geostationary Navigation Overlay Service) has been in operation since 2009.
- GAGAN (GPS Aided Geo Augmented Navigation), for India has been in operation since 2014.
- SDCM (System for Differential Corrections and Monitoring), for Russia is at the time of writing in test mode.

Support of SBAS allows u-blox GPS technology to take full advantage of the augmentation systems that are currently available (i.e. WAAS, EGNOS, MSAS, GAGAN). Signals from systems currently being tested and/or planned (such as SDCM) may also work, when those systems become fully operational, but this cannot be relied upon and u-blox receivers are not configured to support them by default.

With SBAS enabled, the user benefits from additional satellites for ranging (navigation). u-blox GPS technology uses the available SBAS satellites for navigation just like GPS satellites, if the SBAS satellites offer this service.

To improve position accuracy, SBAS uses different types of correction data:

- **Fast Corrections** for short-term disturbances in GPS signals (due to clock problems, etc).
- **Long-term corrections** for GPS clock problems, broadcast orbit errors etc.
- **Ionosphere corrections** for Ionosphere activity

Another benefit of SBAS is the use of GPS integrity information. In this way SBAS control stations can 'disable' the use of GPS satellites within a 6-second alarm time in case of major GPS satellite problems. If integrity monitoring is enabled, u-blox GPS technology only uses satellites, for which integrity information is available.

For more information on SBAS and associated services, refer to the following resources:

- RTCA/DO-229D (MOPS). Available from www.rtca.org
- gps.faa.gov for information on WAAS.
- www.esa.int for information on EGNOS.
- www.essp-sas.eu for information about European Satellite Services Provider (ESSP), the EGNOS operations manager.
- www.isro.org for information on GAGAN.
- www.sdcm.ru for information on SDCM.

SBAS satellites tracked (as of November 2015)

Identification	Position	GPS PRN	SBAS Provider
AMR	98° W	133	WAAS
PanAmSat Galaxy XV	133.0° W	135	WAAS
TeleSat Anik F1R	107.3° W	138	WAAS
Inmarsat 3F2 AOR-E	15.5° W	120	EGNOS
Artemis	21.5° W	124	EGNOS
Inmarsat 3F5 IOR-W	25° E	126	EGNOS
MTSAT-1R	140.1° E	129	MSAS
MTSAT-2	145° E	137	MSAS
Inmarsat-4F1/IOR	64° E	127	GAGAN
GSAT-10	83° E	128	GAGAN

5.2 SBAS Features

⚠ This u-blox SBAS implementation is, in accordance with standard RTCA/DO-229D, a class Beta-1 equipment. All timeouts etc. are chosen for the En Route Case. Do not use this equipment under any circumstances for "safety of life" applications!

u-blox receivers are capable of receiving multiple SBAS signals concurrently, even from different SBAS systems (WAAS, EGNOS, MSAS, etc.). They can be tracked and used for navigation simultaneously. Every tracked SBAS satellite utilizes one vacant receiver tracking channel. Only the number of receiver channels limits the total number of satellites used. Every SBAS satellite that broadcasts ephemeris or almanac information can be used for navigation, just like a normal GPS satellite.

For receiving correction data, the u-blox receiver automatically chooses the best SBAS satellite as its primary source. It will select only one since the information received from other SBAS satellites is redundant and/or could be inconsistent. The selection strategy is determined by the proximity of the satellites, the services offered by the satellite, the configuration of the receiver (Testmode allowed/disallowed, Integrity enabled/disabled) and the signal link quality to the satellite.

If corrections are available from the chosen SBAS satellite and used in the navigation calculation, the DGPS flag is set in the receiver's output protocol messages (see `UBX-NAV-PVT`, `UBX-NAV-SOL`, `UBX-NAV-STATUS`, `UBX-NAV-SVINFO`, NMEA Position Fix Flags description). The message `UBX-NAV-SBAS` provides detailed information about which corrections are available and applied.

The most important SBAS feature for accuracy improvement is Ionosphere correction. The measured data from regional RIMS stations are combined to make a TEC (Total Electron Content) Map. This map is transferred to the receiver via the satellites to allow a correction of the ionosphere error on each received satellite.

Supported SBAS messages

Message Type	Message Content	Source
0(0/2)	Test Mode	All
1	PRN Mask Assignment	Primary
2, 3, 4, 5	Fast Corrections	Primary
6	Integrity	Primary
7	Fast Correction Degradation	Primary
9	Satellite Navigation (Ephemeris)	All
10	Degradation	Primary
12	Time Offset	Primary
17	Satellite Almanac	All
18	Ionosphere Grid Point Assignment	Primary
24	Mixed Fast / Long term Corrections	Primary
25	Long term Corrections	Primary
26	Ionosphere Delays	Primary

Each satellite services a specific region and its correction signal is only useful within that region. Planning is crucial to determine the best possible configuration, especially in areas where signals from different SBAS systems can be received:

Example 1: SBAS Receiver in North America

In the eastern parts of North America, make sure that EGNOS satellites do not take preference over WAAS satellites. The satellite signals from the EGNOS system should be disallowed by using the PRN Mask.

Example 2: SBAS Receiver in Europe

Some WAAS satellite signals can be received in the western parts of Europe, therefore it is recommended that the satellites from all but the EGNOS system should be disallowed using the PRN Mask.

⚠ Although u-blox receivers try to select the best available SBAS correction data, it is recommended to configure them to disallow using unwanted SBAS satellites.

⚠ The EGNOS SBAS system does not provide the satellite ranging function.

5.3 SBAS Configuration

To configure the SBAS functionalities use the UBX proprietary message `UBX-CFG-SBAS` (SBAS Configuration).

SBAS Configuration parameters

Parameter	Description
Mode - SBAS Subsystem	Enabled / Disabled status of the SBAS subsystem. To enable/disable SBAS operation use `UBX-CFG-GNSS`. The field in `UBX-CFG-SBAS` is no longer supported.
Mode - Allow test mode usage	Allow / Disallow SBAS usage from satellites in Test Mode (Message 0)
Services/Usage - Ranging	Use the SBAS satellites for navigation
Services/Usage - Apply SBAS correction data	Combined enable/disable switch for Fast-, Long-Term and Ionosphere Corrections
Services/Usage - Apply integrity information	Use integrity data
Number of tracking channels	Should be set using `UBX-CFG-GNSS`. The field in `UBX-CFG-SBAS` is no longer supported.
PRN Mask	Allows selectively enabling/disabling SBAS satellites (e.g. restrict SBAS usage to WAAS-only).

By default, SBAS is enabled with three prioritized SBAS channels and it will use any received SBAS satellites (except for those in test mode) for navigation, ionosphere parameters and corrections.

6 IMES Description

Indoor MEssaging System (IMES) is an extension to the QZSS specification using ground based beacons that broadcast their location. Its purpose is to allow GNSS users to continue to navigate inside buildings, when they can no longer reliably receive satellite based signals.

☞ Operation of IMES beacons is only allowed within Japan.

☞ u-blox receivers with IMES enabled conform to **IS-QZSS v1.5** and do not support v1.4 or earlier IMES signals. In particular, u-blox receivers rely on the IMES station's carrier frequency being 1575.4282MHz ±0.2ppm as specified in the IMES specification. Transmissions from IMES stations that are not within this frequency range are unlikely to be reliably received. Also the receiver expects the preamble `0x9E` as well as the correct sequence of CNT values as specified by the IS-QZSS.

u-blox receivers report the position information they receive from IMES transmitters directly with

UBX-RXM-IMES. They do not, however, combine this information with navigation solutions derived from satellite signals (reported via various NMEA and UBX-NAV messages). Consequently, the IMES position information may not always be consistent with satellite signal derived position information.

6.1 IMES Features

- **50/250bps Auto-Detection:** Both 50bps and 250bps IMES signals are supported by u-blox receivers. The transmitter's data rate is detected automatically which allows the receiver to even work in a mixed 50bps/250bps IMES environment.
- **Dynamic Tracking Channel Allocation:** The allocation of the tracking channels is done dynamically, in the same way that channels are allocated to other GNSS. If sufficient IMES stations are within reach of the receiver, it will track as many signals as it can up to the value of `maxTrkCh` configured in UBX-CFG-GNSS (8 by default). To reserve a certain number of channels for IMES only (preventing them from being dynamically allocated to other GNSS), set the `resTrkCh` field in UBX-CFG-GNSS accordingly.
- **Data summary:** A summary of all the tracked IMES signals and what position information they are providing is given in the UBX-RXM-IMES message.
- **Raw IMES frames:** The raw IMES subframes received from the IMES stations are reported as they are received with UBX-RXM-SFRBX messages.

7 Navigation Configuration Settings Description

This section relates to the configuration message UBX-CFG-NAV5.

7.1 Platform settings

u-blox receivers support different dynamic platform models (see table below) to adjust the navigation engine to the expected application environment. These platform settings can be changed dynamically without performing a power cycle or reset. The settings improve the receiver's interpretation of the measurements and thus provide a more accurate position output. Setting the receiver to an unsuitable platform model for the given application environment is likely to result in a loss of receiver performance and position accuracy.

Dynamic Platform Models

Platform	Description
Portable	Applications with low acceleration, e.g. portable or writs worn devices. Suitable for most situations.
Stationary	Used in timing applications (antenna must be stationary) or other stationary applications. Velocity restricted to 0 m/s. Zero dynamics assumed.
Pedestrian	Applications with low acceleration and speed, e.g. how a pedestrian would move. Low acceleration assumed.
Automotive	Used for applications with equivalent dynamics to those of a passenger car. Low vertical acceleration assumed.
At sea	Recommended for applications at sea, with zero vertical velocity. Zero vertical velocity assumed. Sea level assumed.
Airborne <1g	Used for applications with a higher dynamic range and greater vertical acceleration than a passenger car. No 2D position fixes supported.

Dynamic Platform Models continued

Platform	Description
Airborne <2g	Recommended for typical airborne environments. No 2D position fixes supported.
Airborne <4g	Only recommended for extremely dynamic environments. No 2D position fixes supported.
Wrist	Reserved. Do not use.

Dynamic Platform Model Details

Platform	Max Altitude [m]	MAX Horizontal Velocity [m/s]	MAX Vertical Velocity [m/s]	Sanity check type	Max Position Deviation
Portable	12000	310	50	Altitude and Velocity	Medium
Stationary	9000	10	6	Altitude and Velocity	Small
Pedestrian	9000	30	20	Altitude and Velocity	Small
Automotive	6000	100	15	Altitude and Velocity	Medium
At sea	500	25	5	Altitude and Velocity	Medium
Airborne <1g	50000	100	100	Altitude	Large
Airborne <2g	50000	250	100	Altitude	Large
Airborne <4g	50000	500	100	Altitude	Large
Wrist	9000	30	20	Altitude and Velocity	Medium

☞ Dynamic platforms designed for high acceleration systems (e.g. airborne <2g) can result in a higher standard deviation in the reported position.

☞ If a sanity check against a limit of the dynamic platform model fails, then the position solution is invalidated. The table above shows the types of sanity checks which are applied for a particular dynamic platform model.

7.2 Navigation Input Filters

The navigation input filters in `UBX-CFG-NAV5` mask the input data of the navigation engine.

☞ These settings are already optimized. Do not change any parameters unless advised by u-blox support engineers.

Navigation Input Filter parameters

Parameter	Description
fixMode	By default, the receiver calculates a 3D position fix if possible but reverts to 2D position if necessary (**Auto 2D/3D**). The receiver can be forced to only calculate 2D (**2D only**) or 3D (**3D only**) positions.
fixedAlt and fixedAltVar	The fixed altitude is used if fixMode is set to 2D only. A variance greater than zero must also be supplied.
minElev	Minimum elevation of a satellite above the horizon in order to be used in the navigation solution. Low elevation satellites may provide degraded accuracy, due to the long signal path through the atmosphere.
cnoThreshNum SVs and cnoThresh	A navigation solution will only be attempted if there are at least the given number of SVs with signals at least as strong as the given threshold.

See also comments in section Degraded Navigation below.

7.3 Navigation Output Filters

The result of a navigation solution is initially classified by the fix type (as detailed in the fixType field of UBX-NAV-PVT message). This distinguishes between failures to obtain a fix at all ("No Fix") and cases where a fix has been achieved, which are further subdivided into specific types of fixes (e.g. 2D, 3D, dead reckoning).

Where a fix has been achieved, a check is made to determine whether the fix should be classified as valid or not. A fix is only valid if it passes the navigation output filters as defined in UBX-CFG-NAV5. In particular, both PDOP and accuracy values must lie below the respective limits.

Valid fixes are marked using the valid flag in certain NMEA messages (see Position Fix Flags in NMEA) and the gnssFixOK flag in UBX-NAV-PVT message.

Important: Users are recommended to check the gnssFixOK flag in the UBX-NAV-PVT or the NMEA valid flag. Fixes not marked valid should not normally be used.

The UBX-NAV-SOL and UBX-NAV-STATUS messages also report whether a fix is valid in their gpsFixOK and GPSfixOk flags. These messages have only been retained for backwards compatibility and users are recommended to use the UBX-NAV-PVT message in preference.

7.3.1 Speed (3-D) Low-pass Filter

The UBX-CFG-ODO message offers the possibility to activate a speed (3-D) low-pass filter. The output of the speed low-pass filter is published in the UBX-NAV-VELNED message (speed field). The filtering level can be set via the UBX-CFG-ODO message (velLpGain field) and must be comprised between 0 (heavy low-pass filtering) and 255 (weak low-pass filtering).

Strictly speaking, the internal filter gain is computed as a function of speed. Therefore, the level as defined in the UBX-CFG-ODO message (velLpGain field) defines the nominal filtering level for speeds below 5m/s.

7.3.2 Course over Ground Low-pass Filter

The UBX-CFG-ODO message offers the possibility to activate a course over ground low-pass filter when the speed is below 8m/s. The output of the course over ground (also named heading of motion 2-D) low-pass filter is published in the UBX-NAV-PVT message (headMot field), UBX-NAV-VELNED message (heading field), NMEA-RMC message (cog field) and NMEA-VTG message (cogt field). The filtering level can be set via the UBX-CFG-ODO message (cogLpGain field) and must be comprised between 0 (heavy low-pass filtering) and 255 (weak low-pass filtering).

The filtering level as defined in the UBX-CFG-ODO message (cogLpGain field) defines the filter gain for speeds below 8m/s. If the speed is higher than 8m/s, no course over ground low-pass filtering is performed.

7.3.3 Low-speed Course Over Ground Filter

The UBX-CFG-ODO message offers the possibility to activate a low-speed course over ground filter (also named heading of motion 2-D). This filter derives the course over ground from position at very low speed. The output of the low-speed course over ground filter is published in the UBX-NAV-PVT message (headMot field), UBX-NAV-VELNED message (heading field), NMEA-RMC message (cog field) and NMEA-VTG message (cogt field). If the low-speed course over ground filter is not activated or inactive, then the course over ground is computed as described in section Freezing the Course Over Ground.

7.4 Static Hold

Static Hold Mode allows the navigation algorithms to decrease the noise in the position output when the velocity is below a pre-defined 'Static Hold Threshold'. This reduces the position wander caused by environmental factors such as multi-path and improves position accuracy especially in stationary applications. By default, static hold mode is disabled.

If the speed drops below the defined 'Static Hold Threshold, the Static Hold Mode will be activated. Once Static Hold Mode has been entered, the position output is kept static and the velocity is set to zero until there is evidence of movement again. Such evidence can be velocity, acceleration, changes of the valid flag (e.g. position accuracy estimate exceeding the Position Accuracy Mask, see also section Navigation Output Filters), position displacement, etc.

The `UBX-CFG-NAV5` message additionally allows for configuration of distance threshold (field staticHoldMaxDist). If the estimated position is farther away from the static hold position than this threshold, static mode will be quit.

7.5 Freezing the Course Over Ground

If the low-speed course over ground filter is deactivated or inactive (see section Low-speed Course over Ground Filter), the receiver derives the course over ground from the GNSS velocity information. If the velocity cannot be calculated with sufficient accuracy (e.g., with bad signals) or if the absolute speed value is very low (under 0.1m/s) then the course over ground value becomes inaccurate too. In this case the course over ground value is frozen, i.e. the previous value is kept and its accuracy is degraded over time. These frozen values will not be output in the NMEA messages `NMEA-RMC` and `NMEA-VTG` unless the NMEA protocol is explicitely configured to do so (see NMEA Protocol Configuration).

7.6 Degraded Navigation

Degraded navigation describes all navigation modes which use less than four Satellite Vehicles (SV).

7.6.1 2D Navigation

If the receiver only has three SVs for calculating a position, the navigation algorithm uses a constant altitude to compensate for the missing fourth SV. When an SV is lost after a successful 3D fix (min. four SVs available), the altitude is kept constant at the last known value. This is called a 2D fix.

> u-blox receivers do not calculate any navigation solution with less than three SVs. Only u-blox Timing products can calculate a timing solution with only one SV when they are in stationary mode.

7.7 Geodetic Coordinate Systems and Ellipsoids

In order to have any useful meaning, the positions reported by a u-blox receiver must be referenced to some coordinate system which defines the origin and, for example, which way is "up". For many reasons, including history, practical autonomy and politics, all the major GNSS define their own theoretical coordinate systems from which they realize a practical reference frame by means of a network of reference points. Specifically:

- GPS uses WGS84
- GLONASS uses PZ90

- Galileo uses GTRF
- BeiDou uses CGCS2000

In practice, the relevant organisations choose to keep their respective frames very close to the International Terrestrial Reference Frame (ITRF), defined and managed by the International Earth Rotation and Reference Systems Service (IERS). However, because the Earth's tectonic plates and even parts of the Earth's core move, new versions of ITRF are defined every few years, generally with changes of the order of a few millimetres. Consequently, the major GNSS occasionally decide that they need to update their reference frames to be better aligned to the latest ITRF. So, for example, GPS switched to WGS84 (G1150) in GPS week 1150 (early 2002) based on ITRF2000, while GLONASS switched from PZ90.02 to PZ90.11 at the end of 2013, based on ITRF2008. The net effect of this, is that all the major GNSS use almost the same reference frame, but there are some small (generally sub-cm) differences between them and these differences occasionally change.

In order to produce positions that can be shown on a map, it is necessary to translate between raw coordinates (e.g. x, y, z) and a position relative to the Earth's surface (e.g. latitude, longitude and altitude) and that requires defining the form of ellipsoid that best matches the shape of the Earth. Historically many different ellipsoid definitions have been used for maps, many of which predate the existence of GNSS and show quite significant differences, leading to discrepencies of as much as 100m in places. Fortunately, most digital maps now use the WGS84 ellipsoid, which is distinct from the WGS84 coordinate system, but defined by the same body.

All u-blox receivers use (the current) version of WGS84 frame as their reference frame, carrying out any necessary corrections internally. What is more, by default, u-blox receivers use the WGS84 ellipsoid and therefore all positions communicated from/to a u-blox receiver will be relative to that. However, users can alter this by specifying their chosen geodetic datum parameters using the `UBX-CFG-DAT` message. The table below indicates the values u-blox recommends for use.

Recommended UBX-CFG-DAT parameters

Ellipsoid	majA	flat	dX	dY	dZ	rotX	rotY	rotZ
WGS84 (default)	6378137.0	298.257223563	0.0	0.0	0.0	0.0	0.0	0.0
PZ90	6378136.0	298.257839303	0.0	0.0	0.0	0.0	0.0	0.0
CGCS2000	6378137.0	298.257227101	0.0	0.0	0.0	0.0	0.0	0.0

Where the receiver is configured to use differential correction data (e.g. via an RTCM stream), as a direct consequence, the receiver's coordinate frame will switch to whatever frame the source of correction data is using.

8 Clocks and Time

8.1 Receiver Local Time

The receiver is dependent on a local oscillator (normally a TCXO or Crystal oscillator) for both the operation of its radio parts and also for timing within its signal processing. No matter what nominal frequency the local oscillator has (e.g. 26 MHz), u-blox receivers subdivide the oscillator signal to provide a 1 kHz reference clock signal, which is used to drive many of the receiver's processes. In particular, the measurement of satellite signals is arranged to be synchronised with the "ticking" of this 1 kHz clock signal.

When the receiver first starts, it has no information about how these clock ticks relate to other time systems; it can only count time in 1 millisecond steps. However, as the receiver derives information from the satellites it is tracking or from aiding messages, it estimates the time that each 1 kHz clock tick takes in the time-base of the relevant GNSS system. In previous generations of u-blox receivers this was always the GPS time-base, but for this generation it could be GPS, GLONASS, Galileo, or BeiDou. This estimate of GNSS time based on the local 1 kHz clock is called **receiver local time**.

As receiver local time is a mapping of the local 1 kHz reference onto a GNSS time-base, it may experience occasional discontinuities, especially when the receiver first starts up and the information it has about the time-base is changing. Indeed after a cold start receiver local time will initially indicate the length of time that the receiver has been running. However, when the receiver obtains some credible timing information from a satellite or aiding message, it will jump to an estimate of GNSS time.

8.2 Navigation Epochs

Each navigation solution is triggered by the tick of the 1 kHz clock nearest to the desired navigation solution time. This tick is referred to as a **navigation epoch**. If the navigation solution attempt is successful, one of the results is an accurate measurement of time in the time-base of the chosen GNSS system, called **GNSS system time**. The difference between the calculated GNSS system time and receiver local time is called the **clock bias** (and the **clock drift** is the rate at which this bias is changing).

In practice the receiver's local oscillator will not be as stable as the atomic clocks to which GNSS systems are referenced and consequently clock bias will tend to accumulate. However, when selecting the next navigation epoch, the receiver will always try to use the 1 kHz clock tick which it estimates to be closest to the desired fix period as measured in GNSS system time. Consequently the number of 1 kHz clock ticks between fixes will occasionally vary (so when producing one fix per second, there will normally be 1000 clock ticks between fixes, but sometimes, to correct drift away from GNSS system time, there will be 999 or 1001).

The GNSS system time calculated in the navigation solution is always converted to a time in both the GPS and UTC time-bases for output.

Clearly when the receiver has chosen to use the GPS time-base for its GNSS system time, conversion to GPS time requires no work at all, but conversion to UTC requires knowledge of the number of leap seconds since GPS time started (and other minor correction terms). The relevant GPS to UTC conversion parameters are transmitted periodically (every 12.5 minutes) by GPS satellites, but can also be supplied to the receiver via the UBX-MGA-GPS-UTC aiding message. By contrast when the receiver has chosen to use the GLONASS time-base as its GNSS system time, conversion to GPS time is more difficult as it requires knowledge of the difference between the two time-bases, but conversion to UTC is easier (as GLONASS time is closely linked to UTC).

Where insufficient information is available for the receiver to perform any of these time-base conversions precisely, pre-defined default offsets are used. Consequently plausible times are nearly always generated, but they may be wrong by a few seconds (especially shortly after receiver start). Depending on the configuration of the receiver, such "invalid" times may well be output, but with flags indicating their state (e.g. the "valid" flags in UBX-NAV-PVT).

☞ u-blox receivers employ multiple GNSS system times and/or receiver local times (in order to support multiple GNSS systems concurrently), so users should not rely on UBX messages that report GNSS system time or receiver local time being supported in future. It is therefore recommended to give preference to those messages that report UTC time.

8.3 iTOW Timestamps

All the main UBX-NAV messages (and some other messages) contain an **iTOW** field which indicates the GPS time at which the navigation epoch occurred. Messages with the same iTOW value can be assumed to have come from the same navigation solution.

Note that iTOW values may not be valid (i.e. they may have been generated with insufficient conversion data) and therefore it is not recommended to use the iTOW field for any other purpose.

☞ The original designers of GPS chose to express time/date as an integer week number (starting with the first full week in January 1980) and a time of week (often abbreviated to TOW) expressed in seconds. Manipulating time/date in this form is far easier for digital systems than the more "conventional" year/month/day, hour/minute/second representation. Consequently, most GNSS receivers use this representation internally, only converting to a more "conventional form" at external interfaces. The iTOW field is the most obvious externally visible consequence of this internal representation.

If reliable absolute time information is required, users are recommended to use the UBX-NAV-PVT navigation solution message which also contain additional fields that indicate the validity (and accuracy in UBX-NAV-PVT) of the calculated times (see also the GNSS Times section below for further messages containing time information).

8.4 GNSS Times

Each GNSS has its own time reference for which detailed and reliable information is provided in the messages listed in the table below.

GNSS Times

Time Reference	Message
GPS Time	UBX-NAV-TIMEGPS
BeiDou Time	UBX-NAV-TIMEBDS
GLONASS Time	UBX-NAV-TIMEGLO
Galileo Time	UBX-NAV-TIMEGAL
UTC Time	UBX-NAV-TIMEUTC

8.5 Time Validity

Information about the validity of the time solution is given in the following form:

- **Time validity**: Information about time validity is provided in the `valid` flags (e.g. `validDate` and `validTime` flags in the UBX-NAV-PVT message). If these flags are set, the time is known and considered as valid for being used. These flags can be found in the GNSS Times table in the GNSS Times section above as well as in the UBX-NAV-PVT message.
- **Time validity confirmation**: Information about confirmed validity is provided in the `confirmedDate` and `confirmedTime` flags in the UBX-NAV-PVT message. If these flags are set, the time validity could be confirmed by using an additional independent source, meaning that the probability of the time to be correct is very high. Note that information about time validity confirmation is only available if the `confirmedAvai` bit in the UBX-NAV-PVT message is set. Check UBX-NAV-PVT which Protocol Version supports this flag.

8.6 UTC Representation

UTC time is used in many NMEA and UBX messages. In NMEA messages it is always reported rounded to the nearest hundredth of a second. Consequently, it is normally reported with two decimal places (e.g. 124923.52). What is more, although compatibility mode (selected using UBX-CFG-NMEA) requires three decimal places, rounding to the nearest hundredth of a second remains, so the extra digit is always 0.

UTC time is is also reported within some UBX messages, such as UBX-NAV-TIMEUTC and UBX-NAV-PVT. In these messages date and time are separated into seven distinct integer fields. Six of these (year, month, day, hour, min and sec) have fairly obvious meanings and are all guaranteed to match the corresponding values in NMEA messages generated by the same navigation epoch. This facilitates simple synchronisation between associated UBX and NMEA messages.

The seventh field is called nano and it contains the number of nanoseconds by which the rest of the time and date fields need to be corrected to get the precise time. So, for example, the UTC time 12:49:23.521 would be reported as: hour: 12, min: 49, sec: 23, nano: 521000000.

It is however important to note that the first six fields are the result of rounding to the nearest hundredth of a second. Consequently the nano value can range from -5000000 (i.e. -5 ms) to +994999999 (i.e. nearly 995 ms).

When the nano field is negative, the number of seconds (and maybe minutes, hours, days, months or even years) will have been rounded up. Therefore, some or all of them will need to be adjusted in order to get the correct time and date. Thus in an extreme example, the UTC time 23:59:59.9993 on 31st December 2011 would be reported as: year: 2012, month: 1, day: 1, hour: 0, min: 0, sec: 0, nano: -700000.

Of course, if a resolution of one hundredth of a second is adequate, negative nano values can simply be rounded up to 0 and effectively ignored.

Which master clock the UTC time is referenced to is output in the message UBX-NAV-TIMEUTC.

The preferred variant of UTC time can be specified using UBX-CFG-NAV5.

8.7 Leap Seconds

Occasionally it is decided (by one of the international time keeping bodies) that, due to the slightly uneven spin rate of the Earth, UTC has moved sufficiently out of alignment with mean solar time (i.e. the Sun no longer appears directly overhead at 0 longitude at midday). A "leap second" is therefore announced to bring UTC back into close alignment. This normally involves adding an extra second to the last minute of the year, but it can also happen on 30th June. When this happens UTC clocks are expected to go from 23:59:59 to 23:59:60 and only then on to 00:00:00.

It is also theoretically possible to have a negative leap second, in which case there will only be 59 seconds in a minute and 23:59:58 will be followed by 00:00:00.

u-blox receivers are designed to handle leap seconds in their UTC output and consequently users processing UTC times from either NMEA and UBX messages should be prepared to handle minutes that are either 59 or 61 seconds long.

Leap second information be be polled with the message UBX-NAV-TIMELS.

8.8 Real Time Clock

u-blox receivers contain circuitry to support a **real time clock**, which (if correctly fitted and powered) keeps time while the receiver is otherwise powered off. When the receiver powers up, it attempts to use the real time clock to initialise receiver local time and in most cases this leads to appreciably faster first fixes.

8.9 Date

All GNSS frequently transmit information about the current time within their data message. In most cases, this is a time of week (often abbreviated to TOW), which indicates the elapsed number of seconds since the start of the week (midnight Saturday/Sunday). In order to map this to a full date, it is necessary to know which week and so the GNSS also transmit a week number, typically every 30 seconds. Unfortunately the GPS data message was designed in a way that only allows the bottom 10 bits of the week number to be transmitted. This is not sufficient to yield a completely unambiguous date as every 1024 weeks (a bit less than 20 years), the transmitted week number value "rolls over" back to zero. Consequently, GPS receivers can't tell the difference between, for example, 1980, 1999 or 2019 etc.

Fortunately, although BeiDou and Galileo have similar representations of time, they transmit sufficient bits for the week number to be unambiguous for the forseeable future (the first ambiguity will be in 2078 for Galileo and not until 2163 for BeiDou). GLONASS has a different structure, based on a time of day, but again transmits sufficient information to avoid any ambiguity during the expected lifetime of the system (the first ambiguous date will be in 2124). Therefore, u-blox 8 / u-blox M8 receivers regard the date information transmitted by GLONASS, BeiDou and Galileo to be unambiguous and, where necessary, use this to resolve any ambiguity in the GPS date.

☞ Customers attaching u-blox receivers to simulators should be aware that GPS time is referenced to 6th January 1980, GLONASS to 1st January 1996, Galileo to 22nd August 1999 and BeiDou to 1st January 2006; the receiver cannot be expected to work reliably with signals that appear to come from before these dates.

8.9.1 GPS-only Date Resolution

In circumstances where only GPS signals are available and for receivers with earlier firmware versions, the receiver establishes the date by assuming that all week numbers must be at least as large as a reference rollover week number. This reference rollover week number is hard-coded into the firmware at compile time and is normally set a few weeks before the s/w is completed, but it can be overridden by the `wknRollover` field of the UBX-CFG-NAVX5 message to any value the user wishes.

The following example illustrates how this works: Assume that the reference rollover week number set in the firmware at compile time is 1524 (which corresponds to a week in calendar year 2009, but would be transmitted by the satellites as 500). In this case, if the receiver sees transmissions containing week numbers in the range 500 ... 1023, these will be interpreted as week numbers 1524 ... 2047 (CY 2009 ... 2019), whereas transmissions with week numbers from 0 to 499 are interpreted as week numbers 2048 ... 2547 (CY 2019 ... 2028).

☞ It is important to set the reference rollover week number appropriately when supplying u-blox receivers with simulated signals, especially when the scenarios are in the past.

9 Broadcast Navigation Data

Reporting of broadcast navigation data is supported for products using protocol version 17 onwards.

The UBX-RXM-SFRBX reports the broadcast navigation data message collected by the receiver from each tracked signal. When enabled, a separate message is generated every time the receiver decodes a complete subframe of data from a tracked signal. The data bits are reported, as received, including preambles and error checking bits as appropriate. However because there is considerable variation in the data structure of the different GNSS signals, the form of the reported data also varies. Indeed, although this document uses the term "subframe" generically, it is not strictly the correct term for all GNSS (e.g. GLONASS has "strings" and Galileo has "pages").

9.1 Parsing Navigation Data Subframes

Each UBX-RXM-SFRBX message contains a subframe of data bits appropriate for the relevant GNSS, delivered in a number of 32 bit words, as indicated by numWords field.

Due to the variation in data structure between different GNSS, the most important step in parsing a UBX-RXM-SFRBX message is to identify the form of the data. This should be done by reading the gnssId field, which indicates which GNSS the data was decoded from. In almost all cases, this is sufficient to indicate the structure and the following sections are organised by GNSS for that reason. However, in some cases the identity of the GNSS is not sufficient, and this is described, where appropriate, in the following sections.

In most cases, the data does not map perfectly into a number of 32 bit words and, consequently, some of the words reported in UBX-RXM-SFRBX messages contain fields marked as "Pad". These fields should be ignored and no assumption should be made about their contents.

UBX-RXM-SFRBX messages are only generated when complete subframes are detected by the receiver and all appropriate parity checks have passed.

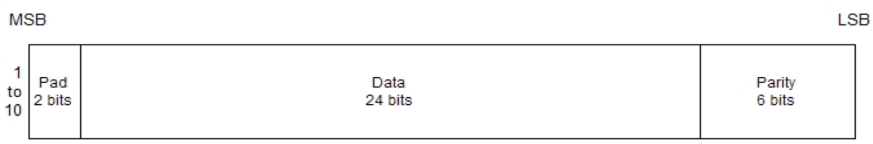
Where the parity checking algorithm requires data to be inverted before it is decoded (e.g. GPS L1C/A), the receiver carries this out before the message output. Therefore, users can process data directly and do not need to worry about repeating any parity processing.

The meaning of the content of each subframe depends on the sending GNSS and is described in the relevant Interface Control Documents (ICD).

9.2 GPS

For GPS (L1C/A) signals, there is a fairly straightforward mapping between the reported subframe and the structure of subframe and words described in the GPS ICD. Each subframe comprises ten data words, which are reported in the same order they are received.

Each word is arranged as follows:



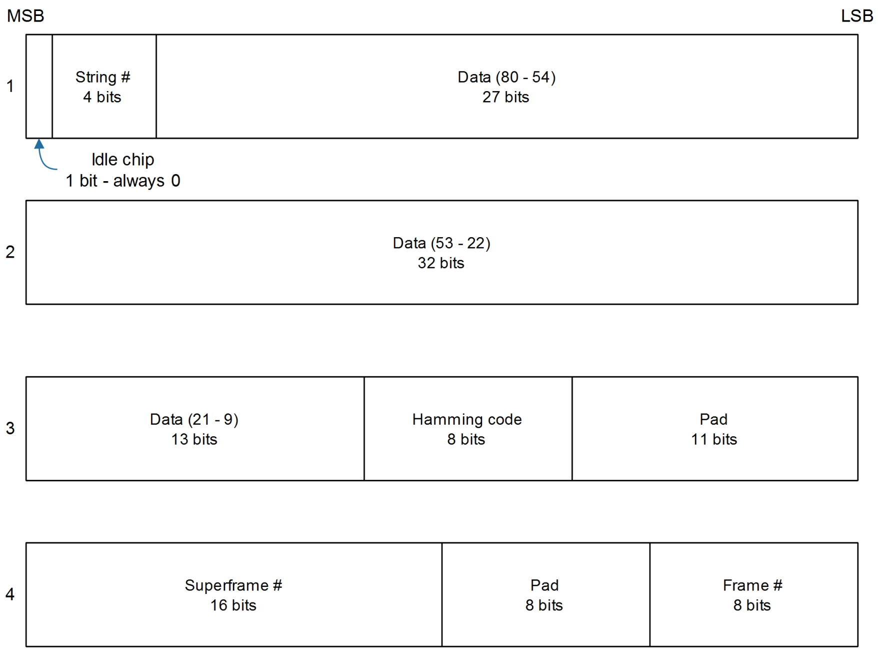
Note that as the GPS data words only comprise 30 bits, the 2 most significant bits in each word reported by UBX-RXM-SFRBX are padding and should be ignored.

9.3 GLONASS

For GLONASS (L1OF) signals, each reported subframe contains a string as described in the GLONASS ICD. This string comprises 85 data bits which are reported over three 32 bit words in the UBX-RXM-SFRBX message. Data bits 1 to 8 are always a hamming code, whilst bits 81 to 84 are a string number and bit 85 is the idle chip, which should always have a value of zero. The meaning of other bits vary with string and frame number.

The fourth and final 32 bit word in the UBX-RXM-SFRBX message contains frame and superframe numbers (where available). These values aren't actually transmitted by the SVs, but are deduced by the receiver and are included to aid decoding of the transmitted data. However, the receiver does not always know these values, in which case a value of zero is reported.

The four words are arranged as follows:

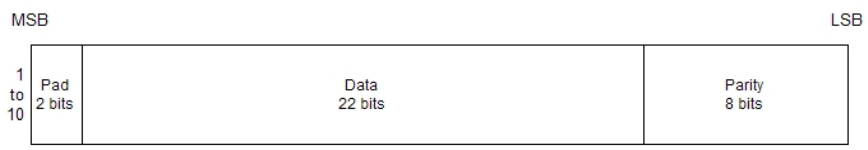


In some circumstances, (especially on startup) the receiver may be able to decode data from a GLONASS SV before it can identify the SV. When this occurs UBX-RXM-SFRBX messages will be issued with an svId of 255 to indicate "unknown".

9.4 BeiDou

For BeiDou (B1I) signals, there is a fairly straightforward mapping between the reported subframe and the structure of subframe and words described in the BeiDou ICD. Each subframe comprises ten data words, which are reported in the same order they are received.

Each word is arranged as follows:

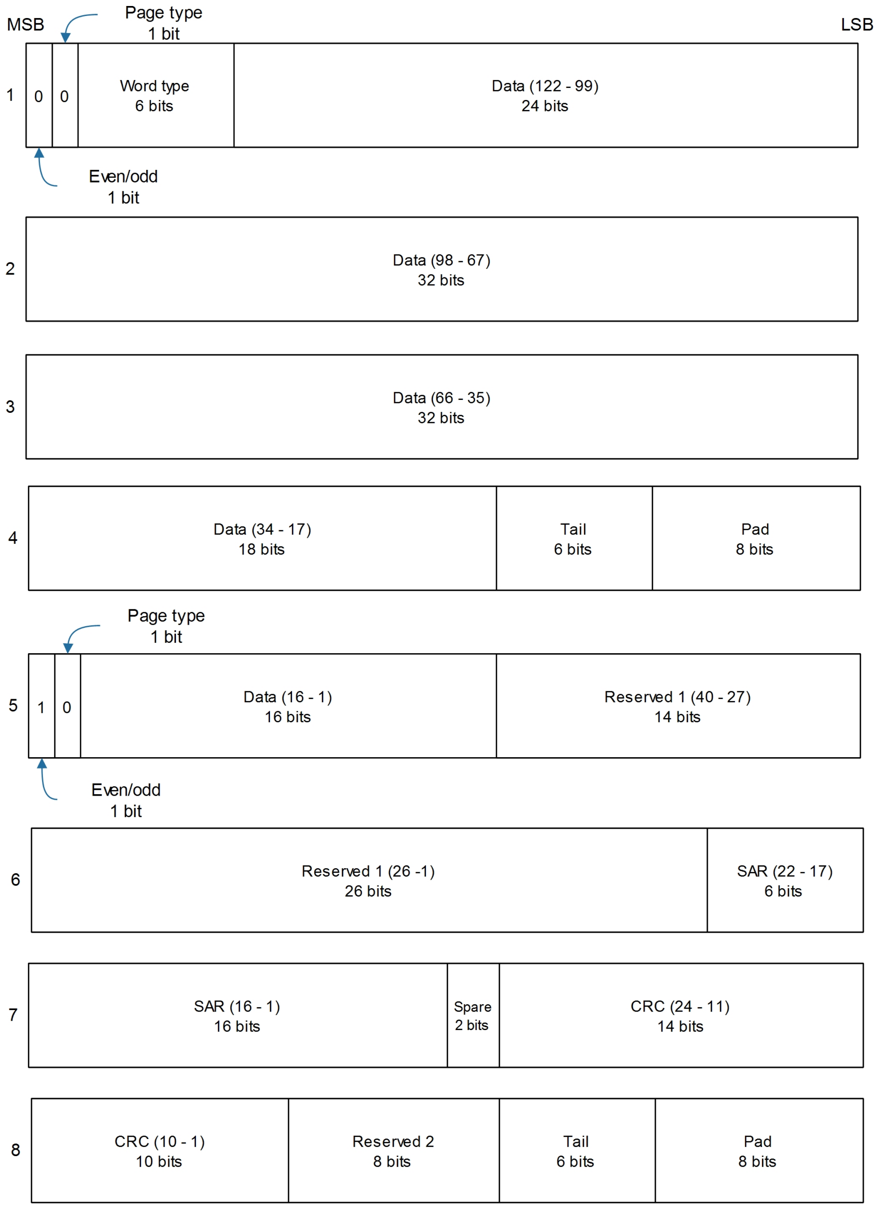


Note that as the BeiDou data words only comprise 30 bits, the 2 most significant bits in each word reported by UBX-RXM-SFRBX are padding and should be ignored.

9.5 Galileo

For Galileo (E1OS) signals, each reported subframe contains a pair of I/NAV pages as described in the Galileo ICD.

Galileo pages can either be "Nominal" or "Alert" pages. For Nominal pages the eight words are arranged as follows:

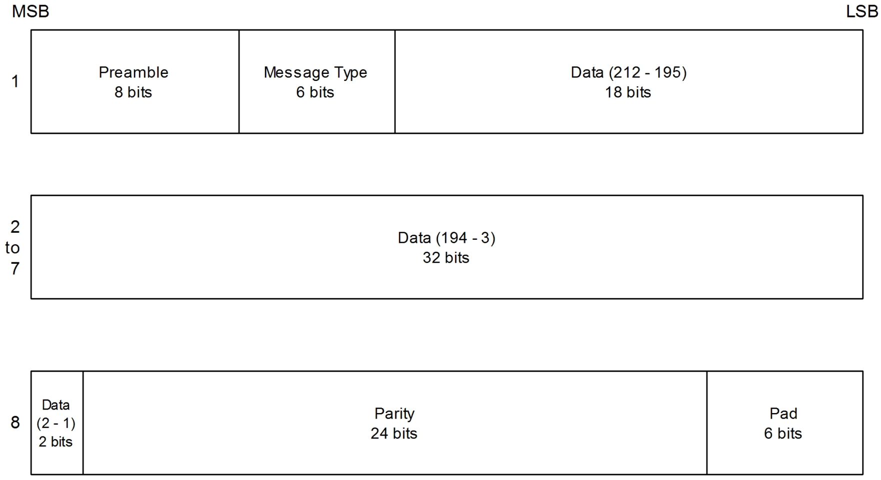


Alert pages are reported in very similar manner, but the page type bits will have value 1 and the structure of the eight words will be slightly different (as indicated by the Galileo ICD).

9.6 SBAS

For SBAS (L1C/A) signals each reported subframe contains eight 32 data words to deliver the 250 bits transmitted in each SBAS data block.

The eight words are arranged as follows:



9.7 QZSS

The structure of the data delivered by QZSS (L1C/A) signals is effectively identical to that for GPS (L1C/A).

The QZSS (L1SAIF) signal is different and uses the same data block format as used by SBAS (L1C/A). QZSS (SAIF) signals can be distinguished from QZSS (L1C/A) by noting that they have 8 words, instead of 10 for QZSS (L1C/A).

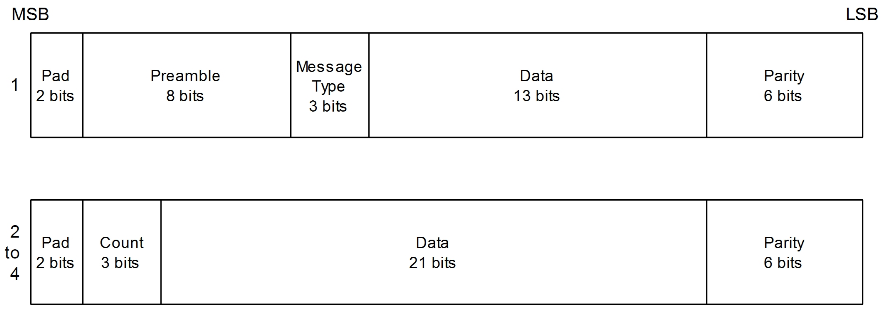
9.8 IMES

Data messages from IMES are of variable length and u-blox receivers currently support the following varieties:

- Short - comprising of a single word
- Medium - comprising of two words
- Position 1 - comprising of three words
- Position 2 - comprising of four words

As a consequence, an IMES `UBX-RXM-SFRBX` message may have a `numWords` value of 1, 2, 3 or 4.

In all cases the structure of words follows the same pattern, with the first word being different from any/all subsequent words as indicated by the following diagram:



9.9 Summary

The following table gives a summary of the different data message formats reported by the `UBX-RXM-SFRBX` message.

Data message formats reported by UBX-RXM-SFRBX

GNSS	Signal	gnssId	numWords	period
GPS	L1C/A	0	10	6s
SBAS	L1C/A	1	8	1s
Galileo	E1OS	2	8	2s
BeiDou	B1I D1	3	10	6s
BeiDou	B1I D2	3	10	0.6s
IMES	Short	4	1	-
IMES	Medium	4	2	-
IMES	Position 1	4	3	-
IMES	Position 2	4	4	-
QZSS	L1C/A	5	10	6s
QZSS	L1SAIF	5	8	1s
GLONASS	L1OF	6	4	2s

10 Serial Communication Ports Description

u-blox receivers come with a highly flexible communication interface. It supports the NMEA and the proprietary UBX protocols, and is truly multi-port and multi-protocol capable. Each protocol (UBX, NMEA) can be assigned to several ports at the same time (multi-port capability) with individual settings (e.g. baud rate, message rates, etc.) for each port. It is even possible to assign more than one protocol (e.g. UBX protocol and NMEA at the same time) to a single port (multi-protocol capability), which is particularly useful for debugging purposes.

To enable a message on a port, the UBX and/or NMEA protocol must be enabled on that port using the UBX proprietary message `UBX-CFG-PRT`. This message also allows changing port-specific settings (baud rate, address etc.). See `UBX-CFG-MSG` for a description of the mechanism for enabling and disabling messages.

The following table shows the port numbers reported in the messages `UBX-MON-IO`, `UBX-MON-MSGPP`, `UBX-MON-TXBUF`, `UBX-MON-RXBUF`. Note that any numbers not listed are reserved for future use.

Port Number assignment

Port #	Electrical Interface
0	DDC (I2C compatible)
1	UART 1
3	USB
4	SPI

10.1 TX-ready indication

This feature enables each port to define a corresponding pin, which indicates if bytes are ready to be transmitted. By default, this feature is disabled. For USB, this feature is configurable but might not behave as described below due to a different internal transmission mechanism. If the number of pending bytes reaches the threshold configured for this port, the corresponding pin will become active (configurable active-low or active-high), and stay active until the last bytes have been transferred from software to hardware (note that this is not necessarily equal to all bytes transmitted, i.e. after the pin has become inactive, up to 16 bytes can still need to be transferred to the host).

The TX-ready pin can be selected from all PIOs which are not in use (see UBX-MON-HW for a list of the PIOs and their mapping), each TX-ready pin is exclusively for one port and cannot be shared. If the PIO is invalid or already in use, only the configuration for the TX-ready pin is ignored, the rest of the port configuration is applied if valid. The acknowledge message does not indicate if the TX-ready configuration is successfully set, it only indicates the successful configuration of the port. To validate successful configuration of the TX-ready pin, the port configuration should be polled and the settings of TX-ready feature verified (will be set to disabled/all zero if the settings are invalid).

The threshold should not be set above 2 kB, as the internal message buffer limit can be reached before this, resulting in the TX-ready pin never being set as messages are discarded before the threshold is reached.

10.2 Extended TX timeout

If the host does not communicate over SPI or DDC for more than approximately 2 seconds, the device assumes that the host is no longer using this interface and no more packets are scheduled for this port. This mechanism can be changed by enabling "extended TX timeouts", in which case the receiver delays idling the port until the allocated and undelivered bytes for this port reach 4 kB. This feature is especially useful when using the TX-ready feature with a message output rate of less than once per second, and polling data only when data is available, determined by the TX-ready pin becoming active.

10.3 UART Ports

One or two Universal Asynchronous Receiver/Transmitter (UART) ports are featured, that can be used to transmit GNSS measurements, monitor status information and configure the receiver. See our online product descriptions for availability.

The serial ports consist of an RX and a TX line. Neither handshaking signals nor hardware flow control signals are available. These serial ports operate in asynchronous mode. The baud rates can be configured individually for each serial port. However, there is no support for setting different baud rates for reception and transmission.

☞ The UART RX interface will be disabled when more than 100 frame errors are detected during a one-second period. This can happen if the wrong baud rate is used or the UART RX pin is grounded. The error message appears when the UART RX interface is re-enabled at the end of the one-second period.

Possible UART Interface Configurations

Baud Rate	Data Bits	Parity	Stop Bits
4800	8	none	1
9600	8	none	1
19200	8	none	1
38400	8	none	1
57600	8	none	1
115200	8	none	1
230400	8	none	1
460800	8	none	1

Note that for protocols such as NMEA or UBX, it does not make sense to change the default word length values (data bits) since these properties are defined by the protocol and not by the electrical interface.

If the amount of data configured is too much for a certain port's bandwidth (e.g. all UBX messages output on a UART port with a baud rate of 9600), the buffer will fill up. Once the buffer space is exceeded, new messages to be sent will be dropped. To prevent message losses, the baud rate and communication speed or the number of enabled messages should be selected so that the expected number of bytes can be transmitted in less than one second.

See UBX-CFG-PRT for UART for a description of the contents of the UART port configuration message.

10.4 How to change between protocols

Reconfiguring a port from one protocol to another is a two-step process:

- Step 1: the preferred protocol(s) needs to be enabled on a port using UBX-CFG-PRT. One port can handle several protocols at the same time (e.g. NMEA and UBX). By default, all ports are configured for UBX and NMEA protocol so in most cases, it's not necessary to change the port settings at all. Port settings can be viewed and changed using the UBX-CFG-PRT messages.
- Step 2: activate certain messages on each port using UBX-CFG-MSG.

11 Multiple GNSS Assistance (MGA)

11.1 Introduction

Users would ideally like GNSS receivers to provide accurate position information the moment they are turned on. With standard GNSS receivers there can be a significant delay in providing the first position fix, principally because the receiver needs to obtain data from several satellites and the satellites transmit that data slowly. Under adverse signal conditions, data downloads from the satellites to the receiver can take minutes, hours or even fail altogether.

Assisted GNSS (A-GNSS) is a common solution to this problem and involves some form of reference network of receivers that collect data such as ephemeris, almanac, accurate time and satellite status and pass this onto to the target receiver via any suitable communications link. Such assistance data enables the receiver to compute a position within a few seconds, even under poor signal conditions.

The UBX-MGA message class provides the means for delivering assistance data to u-blox receivers and customers can obtain it from the u-blox AssistNow Online or AssistNow Offline Services. Alternatively they can obtain assistance data from third-party sources (e.g. SUPL/RRLP) and generate the appropriate UBX-MGA messages to send this data to the receiver.

11.2 Assistance Data

u-blox receivers currently accept the following types of assistance data:

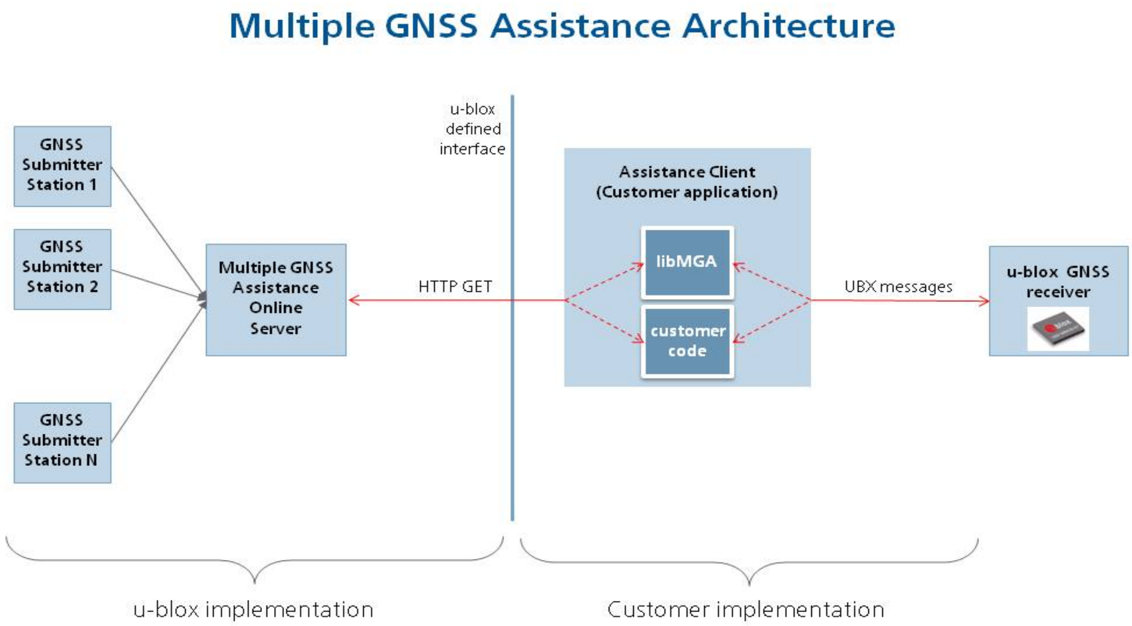
- **Position:** Estimated receiver position can be submitted to the receiver using the UBX-MGA-INI-POS_XYZ or UBX-MGA-INI-POS_LLH messages.
- **Time:** The current time can either be supplied as an inexact value via the standard communication interfaces, suffering from latency depending on the baud rate, or using hardware time synchronization where an accurate time pulse is connected to an external interrupt. The preferred option is to supply UTC time using the UBX-MGA-INI-TIME_UTC message, but times referenced to some GNSS can be delivered with the UBX-MGA-INI-TIME_GNSS message.

- **Clock drift:** An estimate of the clock drift can be sent to the receiver using the UBX-MGA-INI-CLKD message.
- **Frequency:** It is possible to supply hardware frequency aiding by connecting a periodic rectangular signal with a frequency up to 500 kHz and arbitrary duty cycle (low/high phase duration must not be shorter than 50 ns) to an external interrupt, and providing the applied frequency value using the UBX-MGA-INI-FREQ message.
- **Current orbit data:** Each different GNSS transmits orbit data in slightly different forms. For each system there are separate messages for delivering ephemeris and almanac. So for example GPS ephemeris is delivered to the receiver using the UBX-MGA-GPS-EPH message, while GLONASS almanac is delivered with the UBX-MGA-GLO-ALM message.
- **Predicted orbit data:** UBX-MGA-ANO messages can be used to supply predictions of future orbit information to a u-blox receiver. These messages can be obtained from the AssistNow Offline Service and allow a receiver to improve its TTFF even when it is no longer connected to the Internet.
- **Auxiliary information:** Each GNSS transmits some auxiliary data (such as SV health information or UTC parameters) to the receiver. A selection of messages exist for providing such information to the receiver, such as UBX-MGA-GPS-IONO for ionospheric data from GPS.
- **EOP:** Earth Orientation Parameters can be sent to the receiver using the UBX-MGA-INI-EOP message. This will replace the default model used by the AssistNow Autonomous feature and may improve performance (particularly as the receiver gets older and the built-in model decays).
- **Navigation Database:** u-blox receivers can be instructed to dump the current state of their internal navigation database with the UBX-MGA-DBD-POLL message; sending this information back to the receiver (e.g. after a period when the receiver was turned off) restores the database to its former state, and thus allows the receiver to restart rapidly.

11.3 AssistNow Online

AssistNow Online is u-blox' end-to-end Assisted GNSS (A-GNSS) solution for receivers that have access to the Internet. Data supplied by the AssistNow Online Service can be directly uploaded to a u-blox receiver in order to substantially reduce Time To First Fix (TTFF), even under poor signal conditions. The system works by collecting data such as ephemeris and almanac from the satellites through u-blox' Global Reference Network of receivers and providing this data to customers in a convenient form that can be forwarded on directly to u-blox receivers.

The AssistNow Online Service uses a simple, stateless, HTTP interface. Therefore, it works on all standard mobile communication networks that support Internet access, including GPRS, UMTS and Wireless LAN. No special arrangements need to be made with mobile network operators to enable AssistNow Online.



The data returned by the AssistNow Online Service is a sequence of UBX-MGA messages, starting with an estimate of the current time in the form of a UBX-MGA-INI-TIME_UTC message.

☞ AssistNow Online currently supports GPS, GLONASS, BeiDou, Galileo, and QZSS.

☞ Customers may choose to use third party sources of assistance data instead of using the AssistNow Online Service. Customers choosing this option will need to ensure that the data is converted from the format used by the third party source to the appropriate MGA messages. However, it is important to ensure that the receiver has an estimate of the current time before it processes any other assistance data. For this reason, it is strongly recommended to send a UBX-MGA-INI-TIME_UTC or UBX-MGA-INI-TIME_GNSS as the first message of any assistance.

11.3.1 Host Software

As u-blox receivers have no means to connect directly with the Internet, the AssistNow Online system can only work if the host system that contains the receiver can connect to the Internet, download the data from the AssistNow Online Service and forward it on to the receiver. In the simplest case that may involve fetching the data from the AssistNow Online Service (by means of a single HTTP GET request), and sending the resulting data to the receiver.

Depending on the circumstances, it may be beneficial for the host software to include:

- Creating an appropriate UBX-MGA-INI-TIME_UTC message to deliver a better sense of time to the receiver, especially if the host system has a very good sense of the current time and can deliver a time pulse to one of the receiver's EXTINT pins.
- Enable and use flow control to prevent loss of data due to buffer overflow in the receiver.

☞ u-blox provides the source code for an example library, called libMGA, that provides all of the functionality we expect in most host software.

11.3.2 AssistNow Online Sequence

A typical sequence of use of the AssistNow Online Service comprises the following steps:

- Power-up the u-blox receiver
- Request data from the AssistNow Online Service
- Optionally send UBX-MGA-INI-TIME_UTC followed by hardware time synchronization pulse if hardware time synchronization is required.
- Send the UBX messages obtained from the AssistNow Online Service to the receiver.

11.3.3 Flow Control

u-blox receivers aim to process incoming messages as quickly as possible, but there will always be a small delay in processing each message. Uploading assistance data to the receiver can involve sending as many as one hundred of individual messages to the receiver, one after the other. If the communication link is fast, and/or the receiver is busy (trying to acquire new signals), it is possible that the internal buffers will overflow and some messages will be lost. In order to combat this, u-blox receivers support an optional flow control mechanism for assistance.

Flow control is activated by using the configuration item CFG-NAVSPG-ACKAIDING. As a result the receiver will issue an acknowledgement message (UBX-MGA-ACK) for each assistance message it successfully receives. The host software can examine these acknowledgements to establish whether there were any problems with the data sent to the receiver and deduce (by the lack of acknowledgement) if any messages have been lost. It may then be appropriate to resend some of the assistance messages.

The simplest way to implement flow control would be to send one UBX-MGA assistance message at a time, waiting for the acknowledgement, before sending the next. However, such a strategy is likely to introduce significant delays into the whole assistance process. The best strategy will depend on the amount of assistance data being sent and the nature of the communications link (e.g. baud rate of serial link). u-blox recommends that when customers are developing their host software they start by sending all assistance messages and then analyse the resulting acknowledgements to see whether there have been significant losses. Adding small delays during the transmission may be a simple but effective way to avoid substantial loss of data.

11.3.4 Authorization

The AssistNow Online Service is only available for use by u-blox customers. In order to use the services, customers will need to obtain an authorization token from u-blox. This token must be supplied as a parameter whenever a request is made to either service.

11.3.5 Service Parameters

The information exchange with the AssistNow Online Service is based on the HTTP protocol. Upon reception of an HTTP GET request, the server will respond with the required messages in binary format or with an error string in text format. After delivery of all data, the server will terminate the connection.

The HTTP GET request from the client to the server should contain a standard HTTP query string in the request URL. The query string consists of a set of "key=value" parameters in the following form:

key=value;key=value;key=value;

The following rules apply:

- The order of keys is not important.
- Keys and values are case sensitive.
- Keys and values must be separated by an equals character ('=').
- Key/value pairs must be separated by semicolons (';').
- If a value contains a list, each item in the list must be separated by a comma (',').

The following table describes the keys that are supported.

AssistNow Online Parameter Keys

Key Name	Unit/Range	Optional	Description
token	String	Mandatory	The authorization token supplied by u-blox when a client registers to use the service.
gnss	String	Mandatory	A comma separated list of the GNSS for which data should be returned. Valid GNSS are: gps, qzss and glo.
datatype	String	Mandatory	A comma separated list of the data types required by the client. Valid data types are: eph, alm, aux and pos. Time data is always returned for each request. If the value of this parameter is an empty string, only time data will be returned.
lat	Numeric [degrees]	Optional	Approximate user latitude in WGS 84 expressed in degrees and fractional degrees. Must be in range -90 to 90. Example: lat=47.2.
lon	Numeric [degrees]	Optional	Approximate user longitude in WGS 84 expressed in degrees and fractional degrees. Must be in range -180 to 180. Example: lon=8.55.
alt	Numeric [meters]	Optional	Approximate user altitude above WGS 84 Ellipsoid. If this value is not provided, the server assumes an altitude of 0 meters. Must be in range -1000 to 50000.
pacc	Numeric [meters]	Optional	Approximate accuracy of submitted position (see position parameters note below). If this value is not provided, the server assumes an accuracy of 300km. Must be in range 0 to 6000000.
tacc	Numeric [seconds]	Optional	The timing accuracy (see time parameters note below). If this value is not provided, the server assumes an accuracy of 10 seconds. Must be in range 0 to 3600.
latency	Numeric [seconds]	Optional	Typical latency between the time the server receives the request, and the time when the assistance data arrives at the u-blox receiver. The server can use this value to correct the time being transmitted to the client. If this value is not provided, the server assumes a latency of 0. Must be in range 0 to 3600.
filteronpos	(no value required)	Optional	If present, the ephemeris data returned to the client will only contain data for the satellites which are likely to be visible from the approximate position provided by the lat, lon, alt and pacc parameters. If the lat and lon parameters are not provided the service will return an error.
filteronsv	String	Optional	A comma separated list of u-blox gnssId:svId pairs. The ephemeris data returned to the client will only contain data for the listed satellites.

Thus, as an example, a valid parameter string would be:

token=XXXXXXXXXXXXXXXXXXXXXX;gnss=gps,qzss;datatype=eph,pos,aux;lat=47.28;lon=8.56;
pacc=1000

11.3.5.1 Position parameters (lat, lon, alt and pacc)

The position parameters (lat, lon, alt and pacc) are used by the server for two purposes:

- If the filteronpos parameter is provided, the server determines the currently visible satellites at the user position, and only sends the ephemeris data of those satellites which should be in view at the location of the user. This reduces bandwidth requirements. In this case the 'pacc' value is taken into account, meaning that the server will return all SVs visible in the given uncertainty region.
- If the datatype 'pos' is requested, the server will return the position and accuracy in the response data. When this data is supplied to the u-blox receiver, depending on the accuracy of the provided data, the receiver can then choose to select a better startup strategy. For example, if the position is accurate to 100km or better, the u-blox receiver will choose to go for a more optimistic startup strategy. This will result in quicker startup time. The receiver will decide which strategy to choose, depending on the 'pacc' parameter. If the submitted user position is less accurate than what is being specified with the 'pacc' parameter, then the user will experience prolonged or even failed startups.

11.3.5.2 Time parameters (tacc and latency)

Time data is always returned with each request. The time data refers to the time at which the response leaves the server, corrected by an optional latency value.This time data provided by the service is accurate to approximately 10ms but by default the time accuracy is indicated to be +/-10 seconds in order to account for network latency and any time between the client receiving the data and it being provided to the receiver.

If both the network latency and the client latency can safely be assumed to be very low (or are known), the client can choose to set the accuracy of the time message (tacc) to a much smaller value (e.g. 0.5s). This will result in a faster TTFF. The latency can also be adjusted as appropriate. However, these fields should be used with caution: if the time accuracy is not correct when the time data reaches the receiver, the receiver may experience prolonged or even failed start-ups.

For optimal results, the client should establish an accurate sense of time itself (e.g. by calibrating its system clock using a local NTP service) and then modify the time data received from the service as appropriate.

11.3.6 Multiple Servers

u-blox has designed and implemented the AssistNow Online Service in a way that should provide very high reliability. Nonetheless, there will be rare occasions when a server is not available (e.g. due to failure or some form of maintenance activity). In order to protect customers against the impact of such outages, u-blox will run at least two instances of the AssistNow Online Service on independent machines. Customers will have a free choice of requesting assistance data from any of these servers, as all will provide the same information. However, should one fail for whatever reason, it is highly unlikely that the other server(s) will also be unavailable. Therefore customers requiring the best possible availability are recommended to implement a scheme where they direct their requests to a chosen server, but, if that server fails to respond, have a fall-back mechanism to use another server instead.

11.4 AssistNow Offline

AssistNow Offline is a feature that combines special firmware in u-blox receivers and a proprietary service run by u-blox. It is targetted at receivers that only have occasional Internet access and so can't use AssistNow Online. AssistNow Offline speeds up Time To First Fix (TTFF), typically to considerably less than 10s

☞ AssistNow Offline currently supports GPS and GLONASS. u-blox intend to expand the AssistNow Offline Service to support other GNSS (such as BeiDou and Galileo) in due course.

The AssistNow Offline Service uses a simple, stateless, HTTP interface. Therefore, it works on all standard mobile communication networks that support Internet access, including GPRS, UMTS and Wireless LAN. No special arrangements need to be made with mobile network operators to enable AssistNow Offline.

Users of AssistNow Offline are expected to download data from the AssistNow Offline Service, specifying the time period they want covered (1 to 5 weeks) and the types of GNSS. This data must be uploaded to a u-blox receiver, so that it can estimate the positions of the satellites, when no better data is available. Using these estimates will not provide as accurate a position fix as if current ephemeris data is used, but it will allow much faster TTFFs in nearly all cases.

The data obtained from the AssistNow Offline Service is organised by date, normally a day at a time. Consequently the more weeks for which coverage is requested, the larger the amount of data to handle. Similarly, each different GNSS requires its own data and in the extreme cases, several hundred kilobytes of data will be provided by the service. This amount can be reduced by requesting lower resolution, but this will have a small negative impact on both position accuracy and TTFF. See the section on Offline Service Parameters for details of how to specify these options.

The downloaded Offline data is encoded in a sequence of `UBX-MGA-ANO` messages, one for every SV for every day of the period covered. Thus, for example, data for all GPS SVs for 4 weeks will involve in excess of 900 separate messages, taking up around 70kbytes. Where a u-blox receiver has flash storage, all the data can be directly uploaded to be stored in the flash until it is needed. In this case, the receiver will automatically select the most appropriate data to use at any time. See the section on flash-based AssistNow Offline for further details.

AssistNow Offline can also be used where the receiver has no flash storage, or there is insufficient spare flash memory. In this case the customer's system must store the AssistNow Offline data until the receiver needs it and then upload only the appropriate part for immediate use. See the section on host-based AssistNow Offline for further details.

11.4.1 Service Parameters

The information exchange with the AssistNow Offline Service is based on the HTTP protocol. Upon reception of an HTTP GET request, the server will respond with the required messages in binary format or with an error string in text format. After delivery of all data, the server will terminate the connection.

The HTTP GET request from the client to the server should contain a standard HTTP querystring in the request URL. The querystring consists of a set of "key=value" parameters in the following form:

key=value;key=value;key=value;

The following rules apply:

- The order of keys is not important.
- Keys and values are case sensitive.
- Keys and values must be separated by an equals character ('=').
- Key/value pairs must be separated by semicolons (';').
- If a value contains a list, each item in the list must be separated by a comma (',').

The following table describes the keys that are supported.

AssistNow Offline Parameter Keys

Key Name	Unit/Range	Optional	Description
token	String	Mandatory	The authorization token supplied by u-blox when a client registers to use the service.
gnss	String	Mandatory	A comma separated list of the GNSS for which data should be returned. The currently supported GNSS are: gps and glo.
period	Numeric [weeks]	Optional	The number of weeks into the future the data should be valid for. Data can be requested for up to 5 weeks in to the future. If this value is not provided, the server assumes a period of 4 weeks.
resolution	Numeric [days]	Optional	The resolution of the data: 1=every day, 2=every other day, 3=every third day. If this value is not provided, the server assumes a resolution of 1 day.

Thus, as an example, a valid parameter string would be:

token=XXXXXXXXXXXXXXXXXXXXXX;gnss=gps,glo;

11.4.2 Authorization

The AssistNow Offline Service uses the same authorization process as AssistNow Online; see above for details.

11.4.3 Multiple Servers

The AssistNow Offline Service uses the same multiple server mechanism to provide high availability as AssistNow Online; see above for details.

11.4.4 Time, Position and Almanac

While AssistNow Offline can be used on its own, it is expected that the user will provide estimates of the receiver's current position, the current time and ensure that a reasonably up to date almanac is available. In most cases this information is likely to be available without the user needing to do anything. For example, where the receiver is connected to a battery backup power supply and has a functioning real time clock (RTC), the receiver will keep its own sense of time and will retain the last known position and any almanac. However, should the receiver be completely unpowered before startup, then it will greatly improve TTFF if time, position and almanac can be supplied in some form.

Almanac data has a validity period of several weeks, so can be downloaded from the AssistNow Online service at roughly the same time the Offline data is obtained. It can then be stored in the host for uploading on receiver startup, or it can be transferred to the receiver straight away and preserved there (provided suitable non-voltaile storage is available).

Obviously, where a receiver has a functioning RTC, it should be able to keep its own sense of time, but where no RTC is fitted (or power is completely turned off), providing a time estimate via the

UBX-MGA-INI-TIME_UTC message will be beneficial.

Similarly, where a receiver has effective non-volatile storage, the last known position will be recalled, but if this is not the case, then it will help TTFF to provide a position estimate via one of the UBX-MGA-INI-POS_XYZ or UBX-MGA-INI-POS_LLH messages.

Where circumstance prevent the provision of all three of these pieces of data, providing some is likely to be better than none at all.

11.4.5 Flash-based AssistNow Offline

Flash-based AssistNow Offline functionality means that AssistNow Offline data is stored in the flash memory connected to the chip.

The user's host system must download the data from the AssistNow Offline service when an Internet connection is available, and then deliver all of that data to the u-blox receiver. As the total amount of data to be uploaded is large (typically around 100 kbytes) and writing to flash memory is slow, the upload must be done in blocks of up to 512 bytes, one at a time. The UBX-MGA-FLASH-DATA message is used to transmit each block to the receiver.

> AssistNow Offline data stored in flash memory is not affected by any reset of the receiver. The only simple ways to clear it are to completely erase the whole flash memory or to overwrite it with a new set of AssistNow Offline data. Uploading a dummy block of data (e.g. all zeros) will also have the effect of deleting the data, although a small amount of flash storage will be used.

11.4.5.1 Flash-based Storage Procedure

The following steps are a typical sequence for transferring AssistNow Offline data into the receiver's flash memory:

- The host downloads a copy of a latest data from the AssistNow Offline service and stores it locally.
- It sends the first 512 bytes of that data using the UBX-MGA-FLASH-DATA message.
- It awaits a UBX-MGA-FLASH-ACK message in reply.
- Based on the contents of the UBX-MGA-FLASH-ACK message it, sends the next block, resends the last block or aborts the whole process.
- The above three steps are repeated until all the rest of the data has been successfully transferred (or the process has been aborted).
- The host sends an UBX-MGA-FLASH-STOP message to indicate completion of the upload.
- It awaits the final UBX-MGA-FLASH-ACK message in reply. Background processing in the receiver prepares the downloaded data for use at this stage. Particularly if the receiver is currently busy, this maye take quite a few seconds, so the host has to be prepared for a delay before the UBX-MGA-FLASH-ACK is seen.

Note that the final block may be smaller than 512 bytes (where the total data size is not perfectly divisible by 512). Also, the UBX-MGA-FLASH-ACK messages are distinct from the UBX-MGA-ACK messages used for other AssistNow functions.

Any existing data will be deleted as soon as the first block of new data arrives, so no useful data will be available till the completion of the data transfer. Each block of data has a sequence number, starting at zero for the first block. In order to guard against invalid partial data downloads the receiver will not accept blocks which are out of sequence.

11.4.6 Host-based AssistNow Offline

Host-based AssistNow Offline involves AssistNow Offline data being stored until it is needed by the user's host system in whatever memory it has available.

The user's host system must download the data from the AssistNow Offline service when an Internet connection is available, but retain it until the time the u-blox receiver needs it. At this point, the host must upload just the relevant portion of the data to the receiver, so that the receiver can start using it. This is achieved by parsing all the data and selecting for upload to the receiver only those UBX-MGA-ANO messages with a date-stamp nearest the current time. As each is a complete UBX message it can be sent directly to the receiver with no extra packaging. If required the user can select to employ flow control, but in most cases this is likely to prove unnecessary.

When parsing the data obtained from the AssistNow Offline service the following points should be noted:

- The data is made up of a sequence of UBX-MGA-ANO messages
- Customers should not rely on the messages all being a fixed sized, but should read their length from the UBX header to work out where the message ends (and where the next begins).
- Each message indicates the SV for which it is applicable through the svId and gnssId fields.
- Each message contains a date-stamp within the year, month and day fields.
- Midday (UTC) on the day indicated should be considered to be the point at which the data is most applicable.
- The messages will be ordered chronologically, earliest first.
- Messages with same date-stamp will be ordered by ascending gnssId and then ascending svId.

11.4.6.1 Host-based Procedure

The following steps are a typical sequence for host-based AssistNow Offline:

- The host downloads a copy of a latest data from the AssistNow Offline service and stores it locally.
- Optionally it may also download a current set of almanac data from the AssistNow Online service.
- It waits until it want to use the u-blox receiver.
- If necessary it uploads any almanac, position estimate and/or time estimate to the receiver.
- It scans through AssistNow Offline data looking for entries with a date-stamp that most closely matches the current (UTC) time/date.
- It sends each such UBX-MGA-ANO message to the receiver.

Note that when data has been downloaded from the AssistNow Offline service with the (default) resolution of one day, the means for selecting the closest matching date-stamp is simply to look for ones with the current (UTC) date.

11.5 AssistNow Autonomous

11.5.1 Introduction

The assistance scenarios covered by AssistNow Online and AssistNow Offline require an online connection and a host that can use this connection to download aiding data and provide this to the receiver when required.

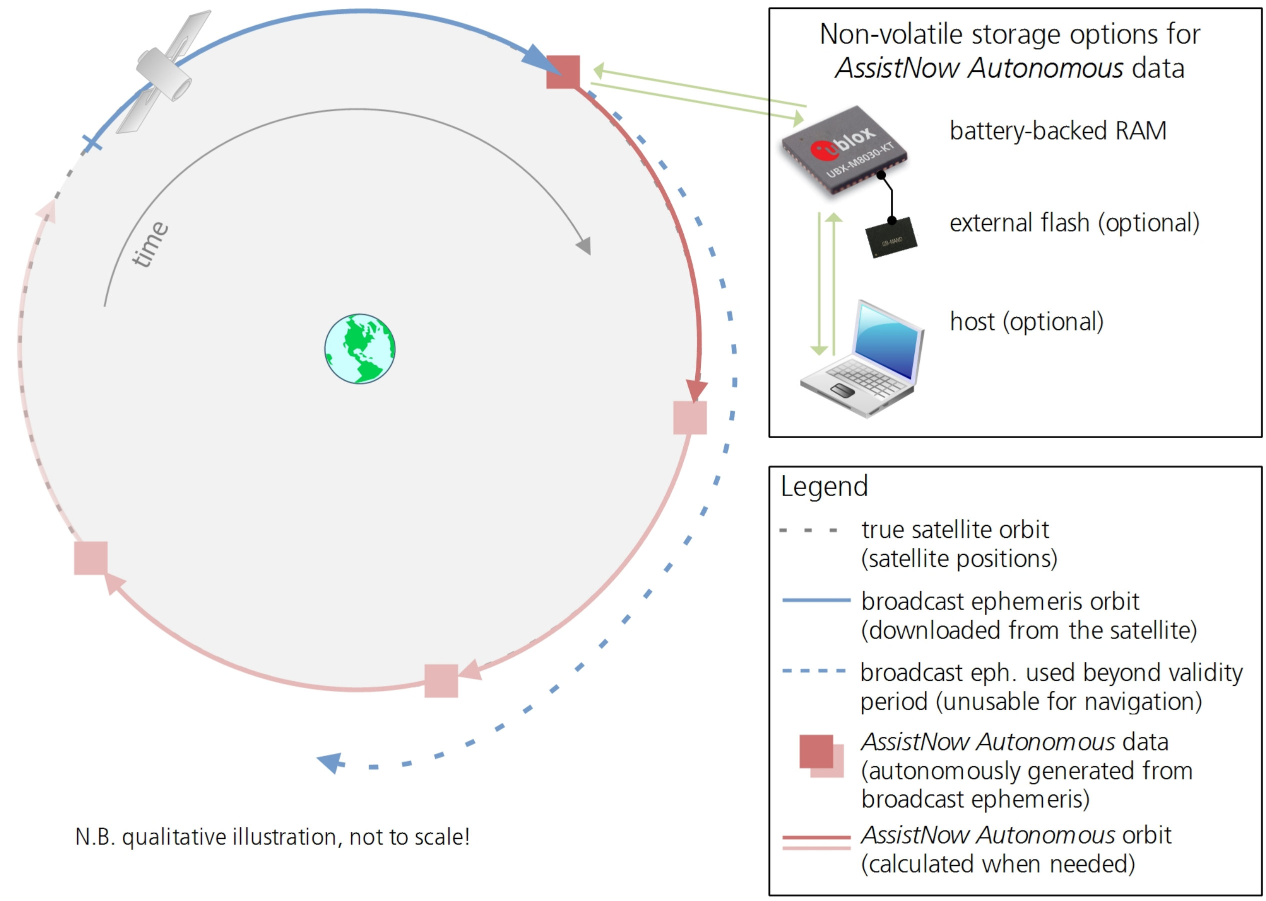
The AssistNow Autonomous feature provides a functionality similar to AssistNow Offline without the need for a host and a connection. Based on a broadcast ephemeris downloaded from the satellite (or obtained by AssistNow Online) the receiver can autonomously (i.e. without any host interaction or online connection) generate an accurate satellite orbit representation («AssistNow Autonomous data») that is usable for navigation much longer than the underlying broadcast ephemeris was intended for. This makes downloading new ephemeris or aiding data for the first fix unnecessary for subsequent start-ups of the receiver.

☞ The AssistNow Autonomous feature is disabled by default. It can be enabled using the UBX-CFG-NAVX5 message.

11.5.2 Concept

The figure below illustrates the AssistNow Autonomous concept in a graphical way. Note that the figure is a qualitative illustration and is not to scale.

- A broadcast ephemeris downloaded from the satellite is a precise representation of a part (for GPS nominally four hours) of the satellite's true orbit (trajectory). It is not usable for positioning beyond this validity period because it diverges dramatically from the true orbit afterwards.
- The AssistNow Autonomous orbit is an extension of one or more broadcast ephemerides. It provides a long-term orbit for the satellite for several revolutions. Although this orbit is not perfectly precise it is a sufficiently accurate representation of the true orbit to be used for navigation.
- The AssistNow Autonomous data is automatically and autonomously generated from downloaded (or assisted) ephemerides. The data is stored automatically in the on-chip battery-backed memory (BBR). Optionally, the data can be backed-up in external flash memory or on the host. The number of satellites for which data can be stored depends on the receiver configuration and may change during operation.
- If no broadcast ephemeris is available for navigation AssistNow Autonomous automatically generates the required parts of the orbits suitable for navigation from the stored data. The data is also automatically kept current in order to minimize the calculation time once the navigation engine needs orbits.
- The operation of the AssistNow Autonomous feature is transparent to the user and the operation of the receiver. All calculations are done in background and do not affect the normal operation of the receiver.
- The AssistNow Autonomous subsystem automatically invalidates data that has become too old and that would introduce unacceptable positioning errors. This threshold is configurable (see below).
- The prediction quality will be automatically improved if the satellite has been observed multiple times. However, this requires the availability of a suitable flash memory (see the Hardware Integration Manual for a list of supported devices). Improved prediction quality also positively affects the maximum usability period of the data.
- AssistNow Autonomous considers GPS, GLONASS, Galileo and BeiDou satellites only. It will not consider satellites on orbits with an eccentricity of >0.05 (e.g., Galileo E18). For GLONASS support a suitable flash memory is mandatory because a single broadcast ephemeris spans to little of the orbit (only approx. 30 minutes) in order to extend it in a usable way. Only multiple observations of the same GLONASS satellite that span at least four hours will be used to generate data.



11.5.3 Interface

Several UBX protocol messages provide interfaces to the AssistNow Autonomous feature. They are:

- The UBX-CFG-NAVX5 message is used to enable or disable the AssistNow Autonomous feature. It is disabled by default. Once enabled, the receiver will automatically produce AssistNow Autonomous data for newly received broadcast ephemerides and, if that data is available, automatically provide the navigation subsystem with orbits when necessary and adequate. The message also allows for a configuration of the maximum acceptable orbit error. See the next section for an explanation of this feature. It is recommended to use the firmware default value that corresponds to a default orbit data validity of approximately three days (for GPS satellites observed once) and up to six days (for GPS and GLONASS satellites observed multiple times over a period of at least half a day).
- Note that disabling the AssistNow Autonomous feature will delete all previously collected satellite observation data from the flash memory.
- The UBX-NAV-AOPSTATUS message provides information on the current state of the AssistNow Autonomous subsystem. The status indicates whether the AssistNow Autonomous subsystem is currently idle (or not enabled) or busy generating data or orbits. Hosts should monitor this information and only power-off the receiver when the subsystem is idle (that is, when the `status` field shows a steady zero).
- The UBX-NAV-SAT message indicates the use of AssistNow Autonomous orbits for individual satellites.

- The UBX-NAV-ORB message indicates the availability of AssistNow Autonomous orbits for individual satellites.
- The UBX-MGA-DBD message provides a means to retrieve the AssistNow Autonomous data from the receiver in order to preserve the data in power-off mode where no battery backup is available. Note that the receiver requires the absolute time (i.e. full date and time) to calculate AssistNow Autonomous orbits. For best performance it is, therefore, recommended to supply this information to the receiver using the UBX-MGA-INI-TIME_UTC message in this scenario.
- The Save-on-Shutdown feature preserves AssistNow Autonomous data.

11.5.4 Benefits and Drawbacks

AssistNow Autonomous can provide quicker start-up times (lower the TTFF) provided that data is available for enough visible satellites. This is particularly true under weak signal conditions where it might not be possible to download broadcast ephemerides at all, and, therefore, no fix at all would be possible without AssistNow Autonomous (or A-GNSS). It is, however, required that the receiver roughly know the absolute time, either from an RTC or from time-aiding (see the Interface section above), and that it knows which satellites are visible, either from the almanac or from tracking the respective signals.

The AssistNow Autonomous orbit (satellite position) accuracy depends on various factors, such as the particular type of satellite, the accuracy of the underlying broadcast ephemeris, or the orbital phase of the satellite and Earth, and the age of the data (errors add up over time).

AssistNow Autonomous will typically extend a broadcast ephemeris for up to three to six days. The UBX-CFG-NAVX5 (see above) message allows changing this threshold by setting the «maximum acceptable modelled orbit error» (in meters). Note that this number does not reflect the true orbit error introduced by extending the ephemeris. It is a statistical value that represents a certain expected upper limit based on a number of parameters. A rough approximation that relates the maximum extension time to this setting is: maxError [m] = maxAge [d] * f, where the factor f is 30 for data derived from satellites seen once and and 16 for data derived for satellites seen multiple time during a long enough time period (see the Concept section above).

There is no direct relation between (true and statistical) orbit accuracy and positioning accuracy. The positioning accuracy depends on various factors, such as the satellite position accuracy, the number of visible satellites, and the geometry (DOP) of the visible satellites. Position fixes that include AssistNow Autonomous orbit information may be significantly worse than fixes using only broadcast ephemerides. It might be necessary to adjust the limits of the Navigation Output Filters.

A fundamental deficiency of any system to predict satellite orbits precisely is unknown future events. Hence, the receiver will not be able to know about satellites that will have become unhealthy, have undergone a clock swap, or have had a manoeuvre. This means that the navigation engine might rarely mistake a wrong satellite position as the true satellite position. However, provided that there are enough other good satellites, the navigation algorithms will eventually eliminate a defective orbit from the navigation solution.

The repeatability of the satellite constellation is a potential pitfall for the use of the AssistNow Autonomous feature. For a given location on Earth the (GPS) constellation (geometry of visible satellites) repeats every 24 hours. Hence, when the receiver «learned» about a number of satellites at some point in time the same satellites will in most places not be visible 12 hours later, and the available AssistNow Autonomous data will not be of any help. Again 12 hours later, however, usable data would be available because it had been generated 24 hours ago.

The longer a receiver observes the sky the more satellites it will have seen. At the equator, and with full sky view, approximately ten (GPS) satellites will show up in a one hour window. After four hours of observation approx. 16 satellites (i.e. half the constellation), after 10 hours approx. 24 satellites (2/3rd of the constellation), and after approx. 16 hours the full constellation will have been observed (and AssistNow Autonomous data generated for). Lower sky visibility reduces these figures. Further away from the equator the numbers improve because the satellites can be seen twice a day. E.g. at 47 degrees north the full constellation can be observed in approx. 12 hours with full sky view.

The calculations required for AssistNow Autonomous are carried out on the receiver. This requires energy and users may therefore occasionally see increased power consumption during short periods (several seconds, rarely more than 60 seconds) when such calculations are running. Ongoing calculations will automatically prevent the power save mode from entering the power-off state. The power-down will be delayed until all calculations are done.

⚠ The AssistNow Offline and AssistNow Autonomous features are exclusive and should not be used at the same time. Every satellite will be ignored by AssistNow Autonomous if there is AssistNow Offline data available for it.

12 Power Management

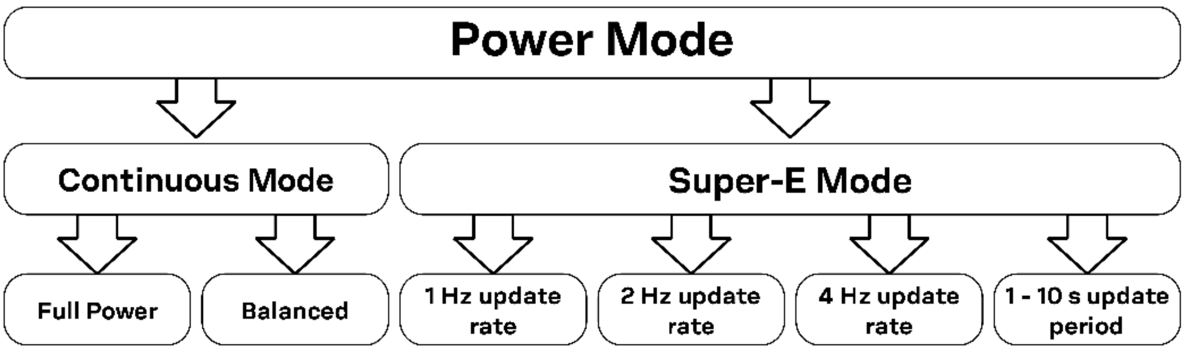
The receiver offers a power-optimized architecture with built-in autonomous power saving functions to minimize power consumption at any given time. The receiver can operate in two power modes:

- Super-Efficient (Super-E) power save mode (PSM) to optimize power consumption.
- Continuous mode for best GNSS reception performance.

The receiver defaults to Super-E mode on power up.

The available power modes are illustrated in figure below. Super-E mode has three predefined settings for 1 Hz (default), 2 Hz and 4 Hz update rates. In addition, Super-E mode supports longer user-defined update periods from 1 second up to 10 seconds. The continuous mode has two predefined settings, full power and balanced.

Power modes



For specific power-saving applications, the host system also has an option to turn the receiver off or to put it into its backup/sleep state. All essential data for quick re-starting of navigation can be saved either on the receiver side or on the host processor side.

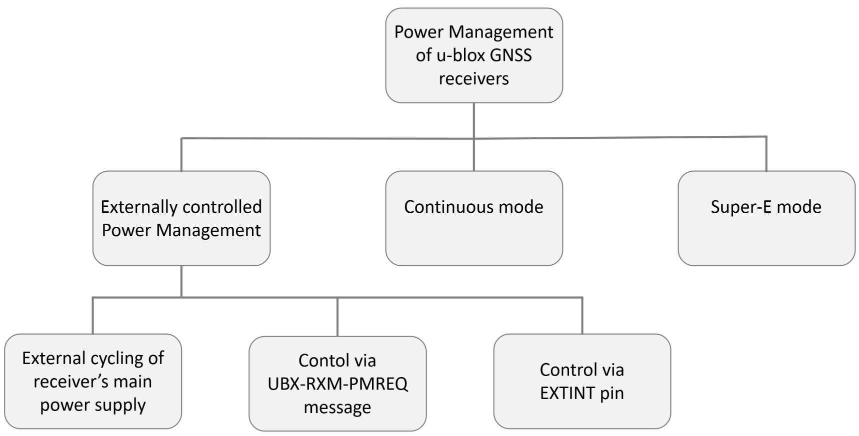
☞ Unlike some other u-blox M8 receivers, the receiver does not support self-managed ON/OFF power saving mode where the receiver periodically puts itself into backup state when an operation interval longer than 10 seconds is selected. The receiver also does not have acquisition timeout logic where the receiver enters backup/sleep state if it cannot acquire any fix in acquisition state.

Receiver power management for power save mode can further be split into two categories:

- Externally controlled power management: This includes various modes of power management for turning the receiver off or putting it to the backup/sleep state. These modes are directly operated by the user or host device: 1. External cycling of the receiver main power supply. 2. Instruct the receiver to enter or exit backup/sleep state via the `UBX-RXM-PMREQ` message. 3. Instruct the receiver to enter or exit backup/sleep state via EXTINT pin
- Internally controlled power management: In Super-E mode the receiver makes the decision when to power down or up some of its internal components according to predefined parameters. In this document Super-E mode is also referred to as power save mode (PSM) or power optimized tracking (POT).

The following figure summarizes the power management modes and their control.

Power management summary



The majority of this section describes Super-E mode (internally controlled power management). However, some the concepts relevant to the externally controlled power management are detailed, such as the EXTINT pin control, Wake up and Power on/off command.

Externally controlled power management operations can be used on top of the internally controlled power management and they do override their operation.

12.1 Super-E power save mode

Super-E mode provides optimal power savings while maintaining a good level of position and speed accuracy.

Super-E mode uses the acquisition engine until a sufficient number of satellites have been acquired for reliable GNSS performance, and uses the tracking engine to track the satellites. The tracking engine is duty-cycled adaptively according to the signal strength in order to provide the best balance between power consumption and navigation performance. The receiver defaults to Super-E mode on power up.

12.1.1 Super-E operation

On receiver start-up, Super-E mode uses the acquisition engine until a sufficient number of satellites have been acquired for reliable GNSS performance, and uses the tracking engine to track the satellites. By default, the acquisition engine is active for at least 5 minutes after the receiver start-up to read the ephemeris of several satellites. The tracking engine is duty-cycled adaptively according to the signal strength, in order to provide the best balance between power consumption and navigation performance.

Super-E mode offers the choice of 1 Hz (default), 2 Hz, or 4 Hz operation. In addition, a slower operation rate with an interval of 1 - 10 seconds can be selected. The higher 2 Hz and 4 Hz navigation rates improve the navigation accuracy, but they also consume more power. The power mode can be selected with the configuration message `UBX-CFG-PMS`. Update periods longer than 1 second are set with the extended power management configuration message `UBX-CFG-PM2`.

Super-E mode has two settings to tune the receiver operation. The "Performance" (default) setting provides the best balance for power vs. performance. The "power save" setting provides up to an additional 15-20% power savings at the cost of position accuracy. The desired setting can be selected by the optTarget configuration option of the extended power management configuration message `UBX-CFG-PM2`.

During the tracking phase of Super-E mode, the satellite reception is duty-cycled and it is turned off most of the time. The receiver reads data from the satellite transmissions only occasionally. Mostly it just checks where the tracked satellites are at that time, and then calculates the position. With a strong enough signal strength, the active time is 1/12 of each navigation cycle. If the signal level falls too low, the active time can increase up to 1/3 of each navigation cycle.

Optimal efficiency of Super-E mode is achieved with a strong signal level. To ensure best efficiency, significant power savings, and good tracking performance, the signal strength of the strongest satellites should be at least 146 dBm to 144 dBm (C/N0 value of 28 dBHz to 30 dBHz). Super-E mode will still work if the signal level goes lower, but efficiency may then degrade.

Some satellites become obscured every now and then when the receiver moves. In Super-E mode, the receiver needs to be able to track at least 6 - 8 satellites constantly for best efficiency. If some of the currently used satellites are not in view, then the receiver can start to use some other known satellite. If too many of the currently known satellites are obscured, the receiver must restart the acquisition engine and stop power-optimized tracking to read the ephemeris data for the new satellites. This acquisition phase lasts only as long as minimally needed.

Navigation performance improves if ephemeris of many more satellites is known beforehand, because the receiver can then use new satellites even if several of the previously used satellites are out of view.

The five-minute (default) initial acquisition period on receiver startup helps to read the ephemeris of many satellites. Ephemeris data can be provided to the receiver also with Assist Now mechanism. If the ephemeris data for many satellites are known, then there is no need to read this data from the satellite transmission. Such preloading of data improves performance especially when the receiver is started in a low signal level environment (e.g. indoors). The initial acquisition period can be adjusted with the extended power management configuration message `UBX-CFG-PM2`. The period can be reduced, e.g. when assistance is used. The minimum value for initial acquisition period is 0 s, which can be used if, for example, valid AssistNow Online data or up to one-day old AssistNow Offline data are available. Depending on the age of the aiding data and GNSS signal conditions, an initial acquisition period up to two or three minutes may be beneficial.

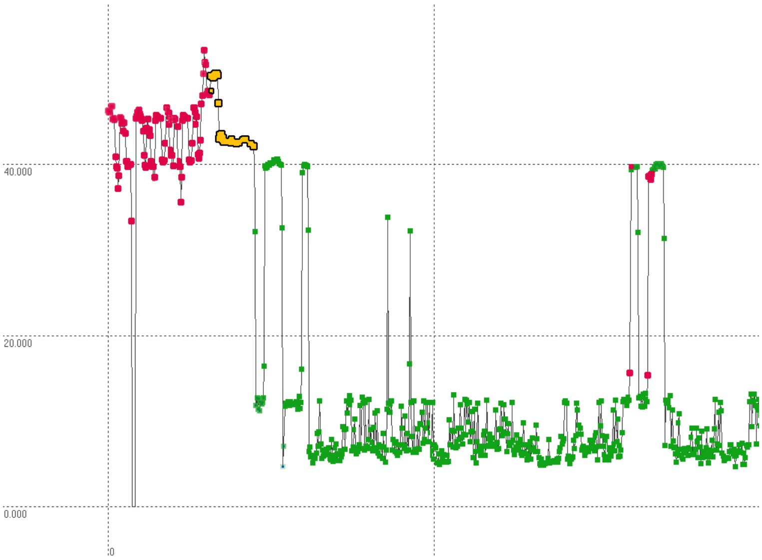
12.1.1.1 Super-E power consumption example

The receiver defaults to Super-E mode on power-up. The receiver starts up in the full-power acquisition state to search for satellites. The acquisition state continues until there is a valid 3D fix and the receiver has enough information about the available satellites. For the 3D fix, the receiver needs to receive data for the current GNSS time and information of at least four satellites (red points in figure below). The receiver continues searching for more satellites in the acquisition state (yellow dots in figure below) until it has enough information for proper low-power operation. By default, this search lasts for five minutes after the receiver start-up, but can be adjusted if, for instance, AssistNow data is used.

After the initial acquisition state, the receiver enters power-optimized tracking state ((shown by the green dots in figure below). This is the low-power state of Super-E mode. If the set of available satellites gets too small, the receiver again enters acquisition or tracking state for a short period until it has enough satellites to track. This is shown by the brief peaks in current consumption during the power-optimized tracking state in figure below.

The state of the receiver is given in the psmState field in the UBX-NAV-PVT message.

Super-E mode power consumption example



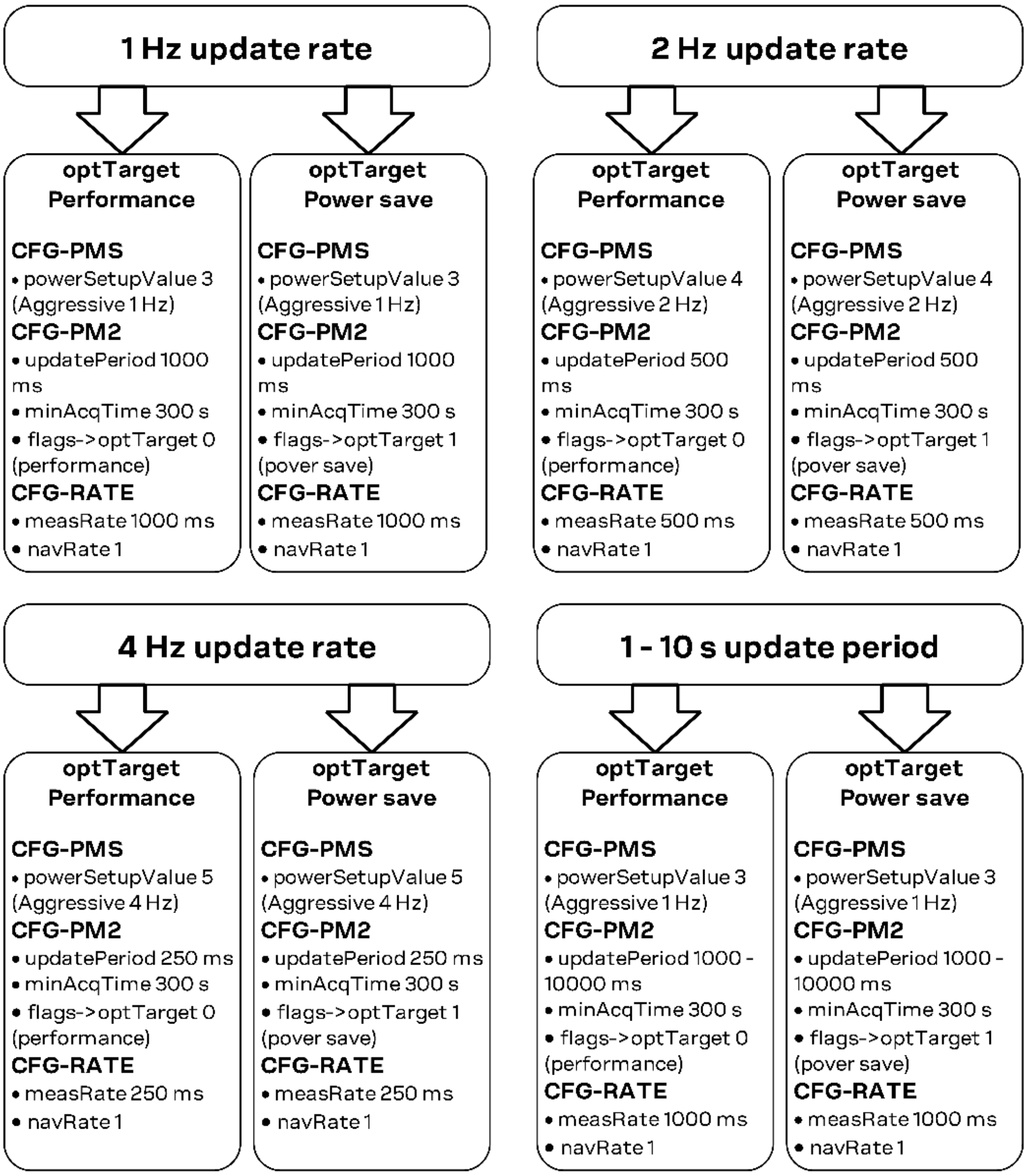
12.1.2 Configuration

Super-E mode and available configuration options are described in detail in figure below. The relevant configuration messages and message fields with required values are also given.

The power mode can be selected with the power mode setup message UBX-CFG-PMS. Super-E mode offers the choice of 1 Hz (default), 2 Hz, or 4 Hz operation. A slower update rate with an interval of 1-10 seconds can be set with the extended power management configuration message UBX-CFG-PM2.

Super-E mode has two settings for tuning the receiver operation. The selection is done with the optTarget configuration option in the extended power management configuration message UBX-CFG-PM2. The "performance" (default) setting provides the best balance for power vs. performance. The "power save" setting provides additional power savings up to 15-20% at the cost of position accuracy.

Super-E mode configuration options



☞ To ensure a consistent receiver configuration, always send the UBX-CFG-PMS message first, followed by the UBX-CFG-PM2 message.

☞ For update rates from 1 Hz to 4 Hz, the update rate in the UBX-CFG-PMS message and the field updatePeriod in UBX-CFG-PM2 must match. For example, for the 2 Hz update rate selected with the UBX-CFG-PMS message, set the updatePeriod in UBX-CFG-PM2 to 500 ms.

☞ For update periods longer than 1 second (up to 10 seconds), first select the 1 Hz update rate with the UBX-CFG-PMS message, followed by the UBX-CFG-PM2 message with the desired value for updatePeriod between 1-10 seconds.

The messages UBX-CFG-PMS and UBX-CFG-PM2 only affect the navigation update rate in the power-optimized tracking state. The update rate for acquisition and tracking states is set with the UBX-CFG-RATE message. For a uniform update rate regardless of the Super-E mode state, the same update rate need to be set with UBX-CFG-PMS/UBX-CFG-PM2 as well as UBX-CFG-RATE messages.

☞ For update rates from 1 Hz to 4 Hz, it is recommended to use a uniform update rate for all of the Super-E mode states.

☞ For longer update periods up to 10 s, it is recommended to set the acquisition and tracking state update rate to 1 Hz with the UBX-CFG-RATE message. This may speed up the return to the power-optimized tracking state in case the receiver needs to enter acquisition or tracking state to decode satellite information.

Super-E mode is designed to only support the operation of GPS/QZSS, GLONASS, and BeiDou. Enabling SBAS or IMES is possible only if at least one of the other systems is enabled. The PSM state behavior will not be altered by enabling SBAS or IMES and it will not take them into account in operation. Therefore, it is recommended to disable them (i.e., SBAS or IMES) when operating in Super-E mode. They can be disabled using UBX-CFG-GNSS.

Note that polling UBX-CFG-PMS will return the setup only if the full configuration is consistent with one of the predefined power mode setups.

☞ Using UBX-CFG-PMS to set Super-E mode 1, 2, or 4 Hz navigation rates sets 180 s minAcqTime instead of the default 300 s. However, 300 s is the recommended value for the best performance.

A number of parameters can be used to customize the power save mode to your specific needs. These parameters are listed in the following table:

Super-E power save mode configuration options on UBX-CFG-PM2

Parameter	Description
updatePeriod	Time between two position fix attempts
minAcqTime	Minimum time the receiver spends in Acquisition state
optTarget	Super-E mode settings will be weighed towards a specific target

☞ Only the settings listed above are relevant for Super-E mode use. Additional power save mode configuration settings supported by other u-blox receivers should not be used.

12.1.2.1 Update period (updatePeriod)

The update period specifies the time between successive position fixes.

12.1.2.2 Minimum acquisition time (minAcqTime)

The receiver tries to obtain a position fix and to download satellite data for at least the time given in minAcqTime. If the receiver determines that it needs more time for the given starting conditions then it will automatically prolong this time. If minAcqTime is set to zero then the minimum acquisition time is exclusively determined by the receiver.

12.1.2.3 Optimization target

In Super-E mode, the behavior of the receiver can be tuned even more closely to the application's need by choosing an appropriate optimization target.

Two optimization targets are available:

- Performance: The receiver achieves a good GNSS performance while keeping the power consumption low.
- Power save: The receiver might sacrifice GNSS performance in favor of a reduced power consumption.

On receiver startup the "performance" setting is selected by default.

For update rates from 1 Hz to 4 Hz, always use UBX-CFG-PMS message to set the update rate. If further configuration with UBX-CFG-PM2 is needed, the field updatePeriod in UBX-CFG-PM2 message must exactly match the update rate set with UBX-CFG-PMS message. For example, if 2 Hz update rate is selected with UBX-CFG-PMS, the field updatePeriod in UBX-CFG-PM2 must be 500 ms.

When the "power save" setting is needed, it must be explicitly set with the UBX-CFG-PM2 message with appropriate value for optTarget.

For update periods longer than 1 s (up to 10 s), first select 1 Hz update rate with UBX-CFG-PMS message, followed by UBX-CFG-PM2 message with the desired value for updatePeriod between 1 - 10 s. For example, to select the power save setting for Super-E mode with 1 Hz navigation rate, the following UBX message is sent to the receiver:

"B5 62 06 3B 30 00 02 06 00 00 02 00 43 01 E8 03 00 00 10 27 00 00 00 00 00 00 00 00 2C 01 2C 01 00 00 CF 40 00 00 87 5A A4 46 FE 00 00 00 20 00 00 00 00 00 00 00 33 74".

The following UBX message restores the "performance" (default) setting for Super-E mode with 1 Hz navigation rate:

"B5 62 06 3B 30 00 02 06 00 00 02 00 43 01 E8 03 00 00 10 27 00 00 00 00 00 00 00 00 2C 01 2C 01 00 00 CF 40 00 00 87 5A A4 46 FE 00 00 00 20 00 00 00 00 00 00 00 33 74".

☞ Sending UBX-CFG-PMS message resets the UBX-CFG-PM2 settings. Always first send UBX-CFG-PMS message followed by UBX-CFG-PM2 if further configuration is needed.

12.2 Continuous mode

Continuous mode provides the best performance in terms of tracking sensitivity and navigation performance by acquiring all satellites that are visible on the sky. continuous mode uses the acquisition engine until all visible satellites are acquired, and uses the tracking engine to track the satellites.

The tracking engine is not duty-cycled in order to achieve the best navigation performance.

If balanced operation is selected for the continuous mode, then some GNSS RF operations are optimized. This reduces the power consumption slightly for the tracking phase.

The navigation update rate in the continuous mode is set with the UBX-CFG-RATE message.

The receiver defaults to Super-E mode on power-up. To use the continuous mode, the operating mode must be either explicitly changed with a UBX message on receiver start-up, or stored as part of the current configuration to an external SQI-flash.

12.3 Backup/sleep state

For specific power-saving applications, the host system also has an option to put the receiver into its backup/sleep state. All essential data for quick re-starting of navigation can be saved either on the receiver side or on the host processor side.

12.3.1 Power on/off command

With message UBX-RXM-PMREQ the receiver can be forced to enter backup/sleep state (in Continuous and power save mode). It will stay in backup/sleep state for the time specified in the message or until it is woken up by an EXTINT or activity on the RXD1, SPI CS, or NRESET pin.

☞ Sending the message UBX-RXM-PMREQ while the receiver is in the power save mode will overrule PSM and force the receiver to enter backup/sleep state. It will stay in backup/sleep state until woken up. After wake-up the receiver continues working in the power save mode as configured.

12.3.2 EXTINT pin control

The operation of the receiver can be externally controlled using either EXTINT0 or EXTINT1 pin. This external control allows the user to decide when to when to force the receiver into sleep/backup mode and when to wake up the receiver.

EXTINT pin control options on UBX-CFG-PM2

Parameter	Description
extintSel	Selects EXTINT pin used with pin control feature
extintBackup	Enables force-OFF pin control feature

The choice of which pin to use can be configured through the extintSelect feature. Only one pin can be selected at a time but it is sufficient to perform all the required tasks.

If the Force-OFF (extintBackup) feature is enabled, setting the configured EXTINT pin to "low" forces the receiver to enter backup/sleep state until next wake up event. Any wake-up event can wake up the receiver even while the pin is set to "low" (see Wake up). However, if the pin stays "low", the receiver will go back to the sleep/backup state immediately. If the pin is "high" the receiver will continue normal operation.

12.3.3 Communication in backup/sleep state

When backup/sleep state is enabled, communication with the receiver (e.g. UBX message to disable this state) requires particular attention. This is because the receiver may be unable to receive any message through its interfaces. To ensure that the configuration messages are processed by the receiver, even while in backup/sleep state, the following steps need to be taken:

- Send a dummy sequence of 0xFF (one byte is sufficient) to the receiver's UART interface. This will wake up the receiver if it is in backup/sleep state. If the receiver is not in backup/sleep state, the sequence will be ignored.
- Send the configuration message about half a second after the dummy sequence. If the interval between the dummy sequence and the configuration message is too short, the receiver may not yet be ready. If the interval is too long, the receiver may return to backup/sleep state before the configuration message was received. It is therefore important to check for a UBX-ACK-ACK reply from the receiver to confirm that the configuration message was received.
- Send the configuration save message immediately after the configuration message.

12.3.4 Wake up signals for backup/sleep state

The receiver can be woken up from backup/sleep state by generating an edge on one of the following pins:

- rising or falling edge on one of the EXTINT pins
- rising or falling edge on the RXD1 pin
- rising or falling edge on the SPI CS pin
- rising edge on NRESET pin

All wake-up signals are interpreted as a position request, where the receiver wakes up and tries to obtain a position fix. Wake-up signals have no effect if the receiver is already in Acquisition, Tracking or POT state.

12.4 Peak current settings

The peak current during acquisition can be reduced by activating the corresponding option in UBX-CFG-PM2. A peak current reduction will result in longer start-up times of the receiver.

> This setting is independent of the activated mode (continuous or power save mode).

12.5 Use cases

Super-E receivers are especially well suited for wearable location tracking applications, but it offers also a wide variety of other uses.

Examples of possible applications are:

- Smartwatches, digital cameras, and other smart portable and wearable devices,
- Sports devices like fitness trackers,
- Trackers for people, pets or assets,
- Safety devices like SOS - Emergency calls, panic buttons.

12.5.1 Occasional location acquisition

Location information may be needed only occasionally, e.g. a timing algorithm of an asset-tracking device decides when a location information is needed, and it can even adjust the timing period based on remaining battery power. A digital camera needs location information for geo-tagging the images and video recordings, but there may be long off-time periods between uses of the device. In such use-cases, the host application should explicitly put the receiver into backup mode or turn it off between the positioning requests to avoid unnecessary power consumption.

Receiver re-start is quick (only a second or two in the best case) and uses only small amount of power if the current navigation information has been saved either on the receiver side or on the host side. Depending on the length of the preceeding power off period, such start is called "hot start" or "warm start".

If previous navigation information has not been saved, the receiver does a "cold start". Cold start operation uses the acquisition engine in the most active mode and can find the location even when overall satellite signal level is quite low. The cold start operation may take quite long time, up to several minutes, in low signal conditions. Speed, sensitivity, accuracy, and power consumption of the cold start operation can be improved by using assistance data. The assistance data can be stored on the receiver side or on optional SQI-flash, or it can be provided to the receiver by the host.

Re-starting the receiver uses more power than keeping it constantly in Super-E mode with the 10 second update period. However, after about two minutes, the overall power consumption of Super-E mode will be greater than the additional power consumption of a warm start or re-start from backup mode. If the application needs location updates less frequently than once per two minutes, then it is beneficial to put the receiver into backup mode or to turn it off during the waiting time.

12.5.2 Location tracking

For continuous location tracking, the receiver should be kept active during the whole tracking session so that it can internally optimize tracking sensitivity, accuracy, and power consumption.

By default, the receiver starts in Super-E mode where it enters low power tracking after the initial acquisition and ephemeris-loading period. Super-E mode provides the best balance between current consumption vs. GNSS performance on location tracking.

Accuracy of tracking can be improved by raising the navigation rate, and power consumption can be improved by lowering the navigation rate. Thus, the navigation rate should be selected for indented purpose.

Super-E mode supports slow navigation rates with fix intervals up to 10 s. If location information is needed less frequently than that, techniques for occasional location acquisition can be used instead of Super-E mode.

In most cases using the receiver in continuous mode instead of Super-E, gives the best GNSS performance for sensitivity and accuracy during location tracking. However, the receiver then consumes more power.

To use the receiver on continuous mode the operating mode must be explicitly changed. There is no low level configuration for continuous mode selection, but the operating mode and length of the initial ephemeris-loading period can be adjusted by sending UBX messages on receiver startup, or by storing these configuration values to an external SQI-flash.

The initial ephemeris-loading period enables good tracking performance even if the receiver must be started with a cold start. However, sensitivity, accuracy and power efficiency of both the first fix acquisition and location tracking can be improved by using assistance data.

Default constellations in the receiver are GPS+QZSS+GLONASS. This default set gives optimal GNSS signal availability and reliability. The set of constellations can be adjusted if needed, e.g. GPS-only mode uses less power, but fix availability and tracking accuracy are not as good as in the default configuration.

13 Forcing a Receiver Reset

Typically, in GNSS receivers, one distinguishes between cold, warm, and hot starts, depending on the type of valid information the receiver has at the time of the restart.

- **Cold start** In cold start mode, the receiver has **no** information from the last position (e.g. time, velocity, frequency etc.) at startup. Therefore, the receiver must search the full time and frequency space, and all possible satellite numbers. If a satellite signal is found, it is tracked to decode the ephemeris (18-36 seconds under strong signal conditions), whereas the other channels continue to search satellites. Once there is a sufficient number of satellites with valid ephemeris, the receiver can calculate position and velocity data. Other GNSS receiver manufacturers call this startup mode `Factory Startup`.
- **Warm start** In warm start mode, the receiver has approximate information for time, position, and coarse satellite position data (Almanac). In this mode, after power-up, the receiver normally needs to download ephemeris before it can calculate position and velocity data. As the ephemeris data usually is outdated after 4 hours, the receiver will typically start with a Warm start if it has been powered down for more than 4 hours. In this scenario, several augmentations are possible. See the section on Multi-GNSS Assistance.
- **Hot start** In hot start mode, the receiver was powered down only for a short time (4 hours or less), so that its ephemeris is still valid. Since the receiver doesn't need to download ephemeris

again, this is the fastest startup method.

In the UBX-CFG-RST message, one can force the receiver to reset and clear data, in order to see the effects of maintaining/losing such data between restarts. For this, the CFG-RST message offers the navBbrMask field, where hot, warm and cold starts can be initiated, and also other combinations thereof.

☞ Data stored in flash memory is not cleared by any of the options provided by UBX-CFG-RST. So, for example, if valid AssistNow Offline data stored in the flash it is likely to have an impact on a "cold start".

The Reset Type can also be specified. This is not related to GNSS, but to the way the software restarts the system.

- **Hardware Reset** uses the on-chip Watchdog, in order to electrically reset the chip. This is an immediate, asynchronous reset. No Stop events are generated. This is equivalent to pull the Reset signal of the receiver to ground.
- **Controlled Software Reset** terminates all running processes in an orderly manner and, once the system is idle, restarts operation, reloads its configuration and starts to acquire and track GNSS satellites.
- **Controlled Software Reset (GNSS only)** only restarts the GNSS tasks, without reinitializing the full system or reloading any stored configuration.
- **Controlled GNSS Stop** stops all GNSS tasks. The receiver will not be restarted, but will stop any GNSS related processing.
- **Controlled GNSS Start** starts all GNSS tasks.

14 Receiver Status Monitoring

Messages in the UBX class UBX-MON are used to report the status of the parts of the embedded computer system that are not GNSS specific.

The main purposes are

- Hardware and Software Versions, using UBX-MON-VER. See also the chapter decoding the output of UBX-MON-VER
- Status of the Communications Input/Output system
- Status of various Hardware Sections with UBX-MON-HW

14.1 Input/Output system

The I/O system is a GNSS-internal layer where all data input- and output capabilities (such as UART, DDC, SPI, USB) of the GNSS receiver are combined. Each communications task has buffers assigned, where data is queued. For data originating at the receiver, to be communicated over one or multiple communications queues, the message UBX-MON-TXBUF can be used. This message shows the current and maximum buffer usage, as well as error conditions.

☞ If the amount of data configured is too much for a certain port's bandwidth (e.g. all UBX messages output on a UART port with a baud rate of 9600), the buffer will fill up. Once the buffer space is exceeded, new messages to be sent will be dropped. For details see section Serial Communication Ports Description

Inbound data to the GNSS receiver is placed in buffers. Usage of these buffers is shown with the message UBX-MON-RXBUF. Further, as data is then decoded within the receiver (e.g. to separate UBX and NMEA data), the UBX-MON-MSGPP can be used. This message shows (for each port and protocol) how many messages were successfully received. It also shows (for each port) how many bytes were discarded because they were not in any of the supported protocol framings.

The following table shows the port numbers used. Note that any numbers not listed are reserved for future use.

Port Number assignment

Port #	Electrical Interface
0	DDC (I2C compatible)
1	UART 1
3	USB
4	SPI

Protocol numbers range from 0-7. All numbers not listed are reserved.

Protocol Number assignment

Protocol #	Protocol Name
0	UBX Protocol
1	NMEA Protocol
2	RTCM Protocol

14.2 Jamming/Interference Indicator

The field `jamInd` of the UBX-MON-HW message can be used as an indicator for continuous wave (narrowband) jammers/interference only. The interpretation of the value depends on the application. It is necessary to run the receiver in an unjammed environment to determine an appropriate value for the unjammed case. If the value rises significantly above this threshold, this indicates that a continuous wave jammer is present.

This indicator is always enabled.

The indicator is reporting any currently detected narrowband interference over all currently configured signal bands

14.3 Jamming/Interference Monitor (ITFM)

The field `jammingState` of the UBX-MON-HW message can be used as an indicator for both broadband and continuous wave (CW) jammers/interference. It is independent of the (CW only) jamming indicator described in Jamming/Interference Indicator above.

This monitor reports whether jamming has been detected or suspected by the receiver. The receiver monitors the background noise and looks for significant changes. Normally, with no interference detected, it will report 'OK'. If the receiver detects that the noise has risen above a preset threshold, the receiver reports 'Warning'. If in addition, there is no current valid fix, the receiver reports 'Critical'.

The monitor has four states as shown in the following table:

Jamming/Interference monitor reported states

Value	Reported state	Description
0	Unknown	Jamming/interference monitor not enabled, uninitialized or antenna disconnected
1	OK	no interference detected

Jamming/Interference monitor reported states continued

Value	Reported state	Description
2	Warning	position ok but interference is visible (above the thresholds)
3	Critical	no reliable position fix and interference is visible (above the thresholds); interference is probable reason why there is no fix

The monitor is disabled by default. The monitor is enabled by sending an appropriate `UBX-CFG-ITFM` message with the `enable` bit set. In this message it is also possible to specify the thresholds at which broadband and CW jamming are reported. These thresholds should be interpreted as the dB level above 'normal'. It is also possible to specify whether the receiver expects an active or passive antenna.

☞ The monitor algorithm relies on comparing the currently measured spectrum with a reference from when a good fix was obtained. Thus the monitor will only function when the receiver has had at least one (good) first fix, and will report 'Unknown' before this time.

The monitor is reporting any currently detected interference over all currently configured signal bands

15 Spoofing Detection

15.1 Introduction

Spoofing is the process whereby someone tries to forge a GNSS signal with the intention of fooling the receiver into calculating a different user position than the true one.

The spoofing detection feature monitors the GNSS signals for suspicious patterns indicating that the receiver is being spoofed. A flag in `UBX-NAV-STATUS` alerts the user to potential spoofing.

15.2 Scope

The spoofing detection feature monitors suspicious changes in the GNSS signal indicating external manipulation. Therefore the detection is only successful when the signal is genuine first and when the transition to the spoofed signal is being observed directly. When a receiver is started up to a spoofed signal the detection algorithms will be unable to recognize the spoofing. Also, the algorithms rely on availability of signals from multiple GNSS; the detection does not work in single GNSS mode.

16 Remote Inventory

16.1 Description

The Remote Inventory enables storing user-defined data in the non-volatile memory of the receiver. The data can be either binary or a string of ASCII characters. In the second case, it will be output at startup after the boot screen.

16.2 Usage

- The contents of the Remote Inventory can be set and polled with the message UBX-CFG-RINV. Refer to the message specification for a detailed description.
- If the contents of the Remote Inventory are polled without having been set before, the default configuration (see table below) is output.

Default configuration

Parameter	Value
flags	0x00
data	"Notice: no data saved!"

☞ As with all configuration changes, these must be saved in order to be made permanent. Make sure to save the section RINV before resetting or switching off the receiver. For more information about saving a configuration, see section Configuration Concept.

17 GNSS time bases

GNSS receivers must handle a variety of different time bases as each GNSS has its own reference system time. What is more, although each GNSS provides a model for converting their system time into UTC, they all support a slightly different variant of UTC. So, for example, GPS supports a variant of UTC as defined by the US National Observatory, while BeiDou uses UTC from the National Time Service Center, China (NTSC). While the different UTC variants are normally closely aligned, they can differ by as much as a few hundreds of nanoseconds.

Although u-blox receivers can combine a variety of different GNSS times internally, the user must choose a single type of GNSS time and, separately, a single type of UTC for input and output.

GNSS signals used in the receiver is selected using UBX-CFG-GNSS message. The UBX-CFG-NAV5 message allows the user to select which variant of UTC the receiver should use. This includes an "automatic" option which causes the receiver to select an appropriate UTC version itself, based on the GNSS configuration, using, in order of preference, USNO if GPS is enabled, SU if GLONASS is enabled, NTSC if BeiDou is enabled and, finally, European if Galileo is enabled. The selected UTC variant is then used in UBX navigation messages from the receiver, and when time is provided to the receiver using the UBX-MGA-INI-TIME_UTC message. Times referenced to some specific GNSS can be delivered with the UBX-MGA-INI-TIME_GNSS message.

☞ u-blox receivers allow users to choose independently GNSS signals used in the receiver and the input/output time base. For example it is possible to instruct the receiver to use GPS and GLONASS satellite signals to generate BeiDou time. This practice will compromise output time accuracy if the receiver cannot measure the timing difference between the constellations directly and is not recommended.

18 Timemark

⚠ In protocol versions 23-23.01 the time mark is supported only in Continuous mode and in acquisition phase of Super-E mode. It is not reliable in the low-power tracking phase of Super-E mode.

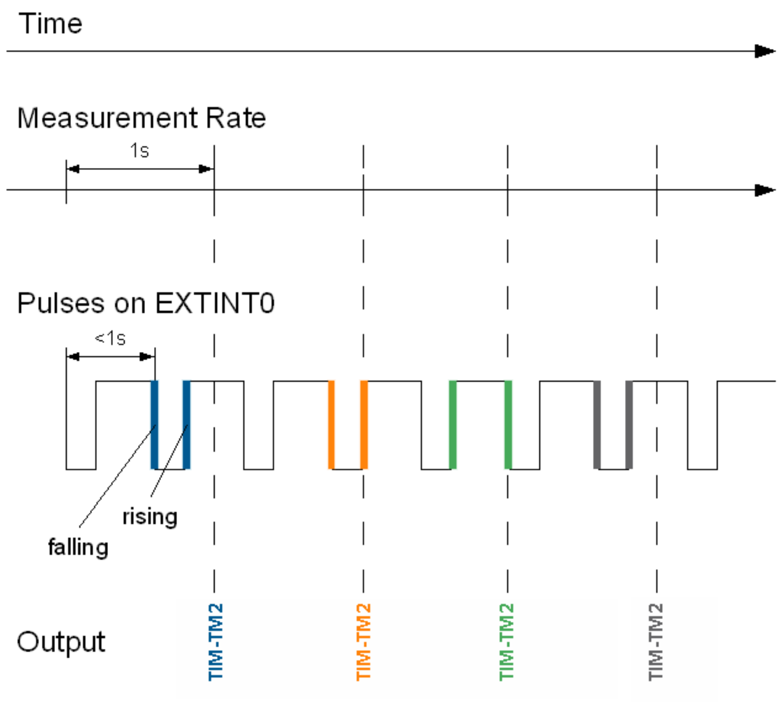
The receiver can be used to provide an accurate measurement of the time at which a pulse was detected on the external interrupt pin. The UTC standard can be set in the UBX-CFG-NAV5 configuration message.

A UBX-TIM-TM2 message is output at the next epoch if

- the UBX-TIM-TM2 message is enabled
- a rising or falling edge was triggered since last epoch on one of the EXTINT channels

The UBX-TIM-TM2 messages include time of the last timemark, new rising/falling edge indicator, time source, validity, number of marks and a quantization error. The timemark is triggered continuously.

☞ Only the last rising and falling edge detected between two epochs is reported since the output rate of the UBX-TIM-TM2 message corresponds to the measurement rate configured with UBX-CFG-RATE (see Figure below).



19 Odometer

19.1 Introduction

The odometer provides information on travelled ground distance (in meter) using solely the position and Doppler-based velocity of the navigation solution. For each computed travelled distance since the last odometer reset, the odometer estimates a 1-sigma accuracy value. The total cumulative ground distance is maintained and saved in the BBR memory.

☞ The odometer feature is disabled by default. It can be enabled using the UBX-CFG-ODO message.

19.2 Odometer Output

The odometer output is published in the UBX-NAV-ODO message. This message contains the following elements:

- Ground distance since last reset (distance field): this distance is defined as the total cumulated distance in meters since the last time the odometer was reset (see section Resetting the Odometer);

- Ground distance accuracy (distanceStd field): this quantity is defined as the 1-sigma accuracy estimate (in meters) associated to the Ground distance since last reset value;
- Total cumulative ground distance (totalDistance field): this quantity is defined as the total cumulated distance in meters since the last time the receiver was cold started (see section Resetting the Odometer).

If logging is enabled, then the odometer's ground distance since last reset value will be included in the logged position data (see section Logging).

19.3 Odometer Configuration

The odometer can be enabled/disabled by setting the appropriate flag in UBX-CFG-ODO (flags field). The algorithm behaviour can be optimized by setting up a profile (odoCfg field) representative of the context in which the receiver is operated. The implemented profiles together with their meanings are listed below:

- Running: the algorithm is optimized for typical dynamics encountered while running, i.e the Doppler-based velocity solution is assumed to be of lower quality;
- Cycling: the algorithm is optimized for typical dynamics encountered while cycling;
- Swimming: the algorithm is optimized for very slow and smooth trajectories typically encountered while swimming;
- Car: the algorithm assumes that good Doppler measurements are available (i.e. the antenna is subject to low vibrations) and is optimized for typical dynamics encountered by cars.

☞ The odometer can only be reliably operated in a swimming context if satellite signals are available and the antenna is not immersed.

19.4 Resetting the Odometer

The odometer outputs (see UBX-NAV-ODO message) can be reset by the following means:

- Ground distance since last reset (distance field): by sending a UBX-NAV-RESETODO message;
- Ground distance accuracy (distanceStd field): by sending a UBX-NAV-RESETODO message;
- Total cumulative ground distance (totalDistance): by a cold start of the receiver (this erases the BBR memory);

20 Logging

20.1 Introduction

The logging feature allows position fixes and arbitrary byte strings from the host to be logged in flash memory attached to the receiver. Logging of position fixes happens independently of the host system, and can continue while the host is powered down.

The following tables list all the logging related messages:

Logging control and configuration messages

Message	Description
UBX-LOG-CREATE	Creates a log file and activates the logging subsystem
UBX-LOG-ERASE	Erases a log file and deactivates the logging subsystem
UBX-CFG-LOGFILTER	Used to start/stop recording and set/get the logging configuration

Logging control and configuration messages continued

Message	Description
UBX-LOG-INFO	Provides information about the logging system
UBX-LOG-STRING	Enables a host process to write a string of bytes to the log file

Logging retrieval messages

Message	Description
UBX-LOG-RETRIEVE	Starts the log retrieval process
UBX-LOG-RETRIEVEPOS	A position log entry returned by the receiver
UBX-LOG-RETRIEVEPOSEXTRA	Odometer position data
UBX-LOG-RETRIEVESTRING	A byte string log entry returned by the receiver
UBX-LOG-FINDTIME	Finds the index of the first entry <= given time

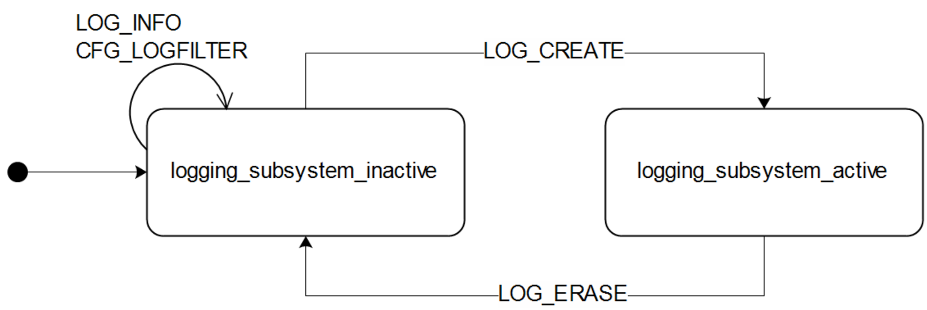
20.2 Setting the logging system up

An empty log can be created using the UBX-LOG-CREATE message and a log can be deleted with the UBX-LOG-ERASE message. The logging system will only be running if a log is in existence, so most logging messages will be rejected with an UBX-ACK-NAK message if there is no log present. Only one log can be created at any one time so an UBX-ACK-NAK message will be returned if a log already exists. The message specifies the maximum size of the log in bytes (with some pre-set values provided). Both the logging subsystem and the receiver file-store have implementation overheads, so total space available for log entries will be somewhat smaller than the size specified.

UBX-LOG-CREATE also allows the log to be specified as a circular log. If the log is circular, then when it fills up, a set of older log entries will be deleted and the space freed up used for new log entries. By contrast, if a non-circular log becomes full then new entries which don't fit will be rejected. UBX-LOG-CREATE also causes the logging system to start up so that further logging messages can be processed. The logging system will start up automatically on power-up if there is a log in existence. The log will remain in the receiver until specifically erased using the UBX-LOG-ERASE message.

UBX-CFG-LOGFILTER controls whether logging of entries is currently enabled and selects position fix messages for logging. These configuration settings will be saved if the configuration is saved to flash. If this is done, then entry logging will continue on power-up in the same manner that it did before power-down.

The top level active/inactive states of the logging subsystem.



20.3 Information about the log

The receiver can be polled for a UBX-LOG-INFO message which will give information about the log. This will include the maximum size that the log can grow to (which, due to overheads, will be smaller than that requested in UBX-LOG-CREATE) and the amount of log space currently occupied. It will also report the number of entries currently in the log together with the time and date of the newest and oldest messages which have a valid time stamp.

Log entries are compressed and have housekeeping information associated with them, so the actual space occupied by log messages may be difficult to predict. The minimum size for a position fix entry is 9 bytes and the maximum 24 bytes, the typical size is 10 or 11 bytes. If the odometer is enabled then this will use at least another three bytes per fix.

Each log also has a fixed overhead which is dependent on the log type. The approximate size of this overhead is shown in the following table.

Log overhead size

Log type	Overhead
circular	Up to 40 kB
non-circular	Up to 8 kB

The number of entries that can be logged in any given flash size can be estimated as follows:

```
Approx. number of entries = (flash size available for logging - log
overhead)/typical entry size
```

For example, if 1500 kB of flash is available for logging (after other flash usage such as the firmware image is taken into account) a non-circular log would be able to contain approximately 139000 entries ((1500*1024)-(8*1024))/11 = 138891.

20.4 Recording

The UBX-CFG-LOGFILTER message specifies the conditions under which entries are recorded. Nothing will be recorded if recording is disabled, otherwise position fix and UBX-LOG-STRING entries can be recorded. When recording is enabled an entry will also be created from each UBX-LOG-STRING message. These will be timestamped if the receiver has current knowledge of time.

The UBX-CFG-LOGFILTER message has several values which can be used to select position fix entries for logging. If all of these values are zero, then all position fixes will be logged (subject to a maximum rate of 1Hz). A position is logged if any of the thresholds are exceeded. If a threshold is set to zero it is ignored. In addition the position difference and current speed thresholds also have a minimum time threshold.

Position fixes are only recorded if a valid fix is obtained - failed and invalid fixes are not recorded.

Position fixes are compressed to economise on the amount of flash space used. In order to improve the compression, the fix values are rounded to improve their compression. This means that the values returned by the logging system may differ slightly from any which are gathered in real time.

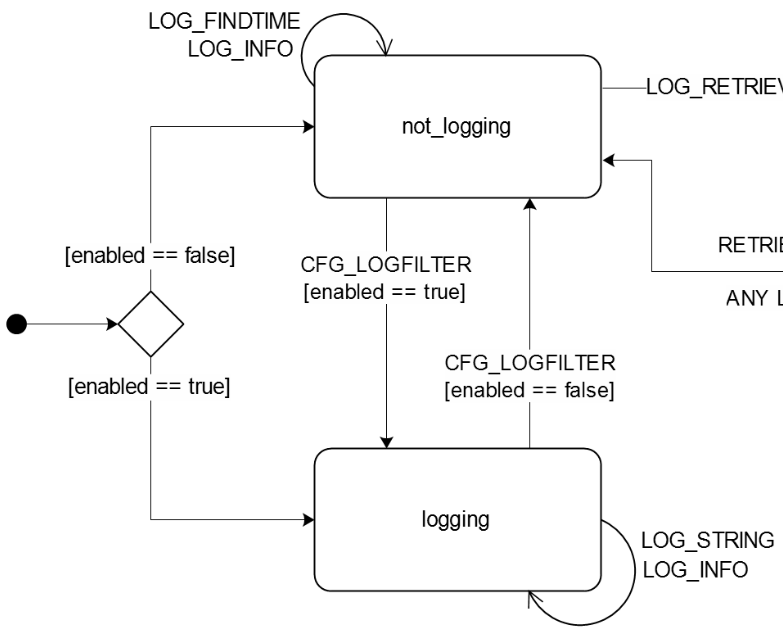
The recorded data for a fix comprises :

- The time and date of the fix recorded to a precision of one second
- Latitude and longitude to a precision of one millionth of a degree. Depending on position on Earth this is a precision in the order of 0.1m
- Altitude (height above mean sea level) to a precision of 0.1m. Entries with an altitude lower than

-470m (lower than the lowest point on earth) or higher than 20,000m may not be recorded in the log.

- Ground speed to a precision of 1cm/s
- The fix type (only successful fix types, since these are the only ones recorded)
- The number of satellites used in the fix is recorded, but there is a maximum count which can be recorded. If the actual count exceeds this maximum count then the maximum count will be recorded. If a log entry is retrieved with a satellite count equal to the maximum this means that value or more. The maximum count is 19.
- A horizontal accuracy estimate is recorded to give an indication of fix quality. This is an approximate compressed representation of the accuracy as determined by the fix process. Any accuracy less than 0.7m will be recorded as 0.7m and any value above 1km will be recorded as 1km. Within these limits, the recorded accuracy will always be greater than the fix accuracy number (by up to 40%)
- Heading to a precision of one degree
- Odometer distance data (if odometer is enabled)

The states of the active logging subsystem



20.5 Retrieval

UBX-LOG-RETRIEVE starts the process which allows the receiver to output log entries. Log recording must be stopped using UBX-CFG-LOGFILTER before this can be done. UBX-LOG-INFO may be helpful to a host system in order to understand the current log status before retrieval is started.

Once retrieval has started, one message will be output from the receiver for each log entry requested. Sending any logging message to the receiver during retrieval will cause the retrieval to stop before the message is processed.

To maximise the speed of transfer it is recommended that a high communications data rate is used and GNSS processing is stopped during the transfer (see UBX-CFG-RST)

UBX-LOG-RETRIEVE can specify a start-entry index and entry-count. The maximum number of entries that can be returned in response to a single UBX-LOG-RETRIEVE message is 256. If more entries than this are required the message will need to be sent multiple times with different startEntry indices.

The receiver will send a UBX-LOG-RETRIEVEPOS message for each position fix log entry and a UBX-LOG-RETRIEVESTRING message for each string log entry. If the odometer was enabled at the time a position was logged, then a UBX-LOG-RETRIEVEPOSEXTRA will also be sent. Messages will be sent in the order in which they were logged, so UBX-LOG-RETRIEVEPOS and UBX-LOG-RETRIEVESTRING messages may be interspersed in the message stream.

The UBX-LOG-FINDTIME message can be used to search a log for the index of the first entry less than or equal to the given time. This index can then be used with the UBX-LOG-RETRIEVE message to provide time-based retrieval of log entries.

20.6 Command message acknowledgement

Some log operations may take a long time to execute because of the time taken to write to flash memory. The time for some operations may be unpredictable since the number and timing of flash operations may vary. In order to allow host software to synchronise to these delays logging messages will always produce a response. This will be UBX-ACK-NAK in case of error, otherwise UBX-ACK-ACK unless there is some other defined response to the message.

It is possible to send a small number of logging commands without waiting for acknowledgement, since there is a command queue, but this risks confusion between the acknowledgements for the commands. Also a command queue overflow would result in commands being lost.

21 Data Batching

21.1 Introduction

The data batching feature allows position fixes to be stored in the RAM of the receiver to be retrieved later in one batch. Batching of position fixes happens independently of the host system, and can continue while the host is powered down.

The following tables list all the batching related messages:

Batching control and configuration messages

Message	Description
UBX-CFG-BATCH	Used to enable and configure the batching feature
UBX-MON-BATCH	Provides information about the buffer fill level and dropped data due to overrun

Batch retrieval messages

Message	Description
UBX-LOG-RETRIEVEBATCH	Starts the batch retrieval process
UBX-LOG-BATCH	A batch entry returned by the receiver

21.2 Setting up the data batching

Data batching is disabled per default and it has to be configured before use via UBX-CFG-BATCH.

The feature must be enabled and the buffer size must be set to greater than 0. It is possible to set up a PIO as a flag that indicates when the buffer is close to filling up. The fill level when this PIO is asserted can be set by the user separately from the buffer size. The notification fill level must not be larger than the buffer size.

If the host does not retrieve the batched fixes before the buffer fills up the oldest fix will be dropped and replaced with the newest.

The RAM available in the chip limits the size of the buffer. To make the best use of the available space users can select what data they want to batch. When batching is enabled a basic set of data is stored and the configuration flags `extraPvt` and `extraOdo` can be used to store more detailed information about the position fixes. Doing so reduces the number of fixes that can be batched.

The receiver will reject configuration if it cannot allocate the required buffer memory. To ensure robust operation of the receiver the following limits are enforced:

Maximum number of batched epochs

extraPvt	extraOdo	Maximum number of epochs
0	0	300
0	1	221
1	0	156
1	1	132

☞ It is recommended to disable all periodic output messages when using batching. This improves system robustness and also helps ensure that the output of batched data is not delayed by other messages.

☞ The buffer size is set up in terms of navigation epochs. This means that the time that can be covered with a certain buffer depends on the navigation rate. This rate can be set separately for full power operation via UBX-CFG-RATE and for power save mode via the updatePeriod in UBX-CFG-PM2.

21.3 Retrieval

UBX-LOG-RETRIEVEBATCH starts the process which allows the receiver to output batch entries. Batching must not be stopped for readout; all batched data is lost when the feature is disabled.

Batched fixes are always retrieved starting with the oldest fix in the buffer and progressing towards newer ones. There is no way to skip certain fixes during retrieval.

When a UBX-LOG-RETRIEVEBATCH message is sent the receiver transmits all batched fixes. It is recommended to send a retrieval request with `sendMonFirst` set. This way the receiver will send a UBX-MON-BATCH message first that contains the number of fixes in the batching buffer. This information can be used to detect when the u-blox receiver finished sending data.

Once retrieval has started, the receiver will first send UBX-MON-BATCH if `sendMonFirst` option was selected in the UBX-LOG-RETRIEVEBATCH. After that, it will send UBX-LOG-BATCH messages with the batched fixes.

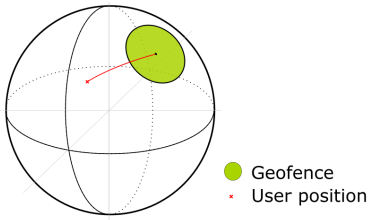
To maximise the speed of transfer it is recommended that a high communications data rate is used.

☞ The receiver will discard retrieval request while processing a previous UBX-LOG-RETRIEVEBATCH message.

☞ The receiver does **not** acknowledge the reception of UBX-LOG-RETRIEVEBATCH; the response that the host should expect are the reply messages.

22 Geofencing

22.1 Introduction



The geofencing feature allows for the configuration of up to four circular areas (geofences) on the Earth's surface. The receiver will then evaluate for each of these areas whether the current position lies within the area or not and signal the state via UBX messaging and PIO toggling.

22.2 Interface

Geofencing can be configured using the UBX-CFG-GEOFENCE message. The geofence evaluation is active whenever there is at least one geofence configured.

The current state of each geofence plus the combined state is output in UBX-NAV-GEOFENCE with every navigation epoch.

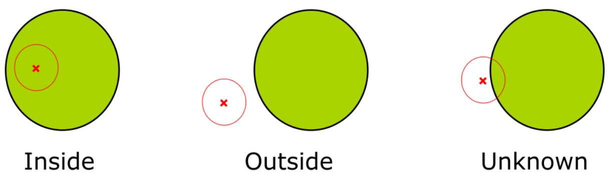
Additionally the user can configure the receiver to output the combined geofence state on a physical pin.

22.3 Geofence state evaluation

With every navigation epoch the receiver will evaluate the current solution's position versus the configured geofences. There are three possible outcomes for each geofence:

- Inside - The position is inside the geofence with the configured confidence level
- Outside - The position lies outside of the geofence with the configured confidence level
- Unknown - There is no valid position solution or the position uncertainty does not allow for unambiguous state evaluation

The position solution uncertainty (standard deviation) is multiplied with the configured confidence sigma level number and taken into account when evaluating the geofence state (red circle in figure below).



The combined state for all geofences is evaluated as the combination (logical OR) of all geofences:

- Inside - The position lies inside of at least one geofence
- Outside - The position lies outside of all geofences
- Unknown - All remaining states

22.4 Using a PIO for Geofence State Output

This feature can be used for example for waking up a sleeping host when a defined geofence condition is reached. The receiver will toggle the assigned pin according to the combined geofence state. Due to hardware restrictions the unknown state will always be represented as HIGH. If the receiver is in software backup or in a reset, the pin will go to HIGH accordingly. The meaning of the LOW state can be configured using UBX-CFG-GEOFENCE.

23 Host Interface Signature Description

23.1 Introduction

The host interface signature feature is designed to help to detect 3rd party attempts to tamper with position and/or time in the host communication channel (i.e. UART).

The level of security of such mechanism depends on how the final system is designed. The feature itself cannot guarantee that the system is secure if the host, the final system HW, and the production setup are not secure.

The feature works by the receiver calculating a numerical signature for the configured messages. The system receiving the message can verify the signature based on the message content and the configured value, termed "seed".

Two new messages are provided for configuring the seed used for the signing: UBX-CFG-FIXSEED and UBX-CFG-DYNSEED.

23.2 Configuring the Fixed Seed and Register Messages

In the UBX-CFG-FIXSEED message the fixed seed and the set of UBX messages to be signed can be configured.

☞ At least one message has to be registered and a maximum of 10 messages are supported.

☞ Configuring the set of messages that are signed will not enable these messages by default.

☞ All UBX messages can be signed.

☞ This message can only be sent once to the receiver. All subsequent messages will result in a NAK answer.

23.3 Configuring the Dynamic Seed

In the UBX-CFG-DYNSEED message an additional seed can be configured to make a replay attack more difficult. This form of attack stores the messages received from the receiver for a certain time and replays them later.

To prevent such an attack the host can use the time information from the receiver or a dynamic seed. This generates a random seed at regular intervals that is then used by the received to sign the outgoing messages.

The frequency of the update on the dynamic seed has to be configured depending on the security concept of the whole system. In case the interval is too long the attacker can store the first set of messages and replay them during the whole period until a new seed is generated. The recommended interval would be in the range of some seconds to a few minutes.

☞

> By default the dynamic seed is set to 0x0000_0000_0000_0000.
>
> While programming the dynamic seed the receiver may send still send signatures which are based on the old seed.

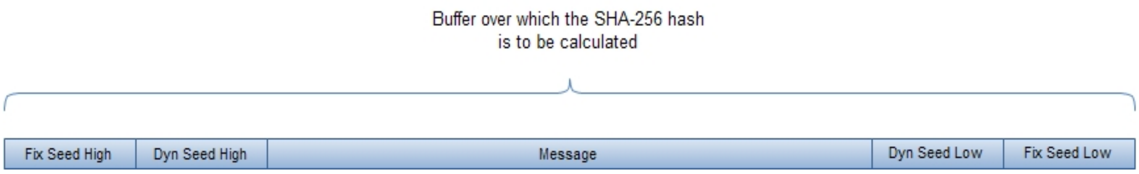
23.4 Parsing the Signature

The UBX-SEC-SIGN message contains the signature of a previously transmitted message and is **always** sent after the related message. It is not guaranteed that between the message and the signature no other messages are output.

The payload of UBX-SEC-SIGN contains the reference to the signed message. It can be used to match the related message using the class ID, the message ID and the UBX checksum of the related message. This means that a previously transmitted message is signed when the class ID, the message ID and the UBX checksum match.

23.5 Calculate the Hash

The picture below shows the layout of the buffer over which the SHA-256 hash is calculated.



The result is a 256 bit (32 bytes) hash which needs to be verified with the content (field hash) of the corresponding UBX-SEC-SIGN message.

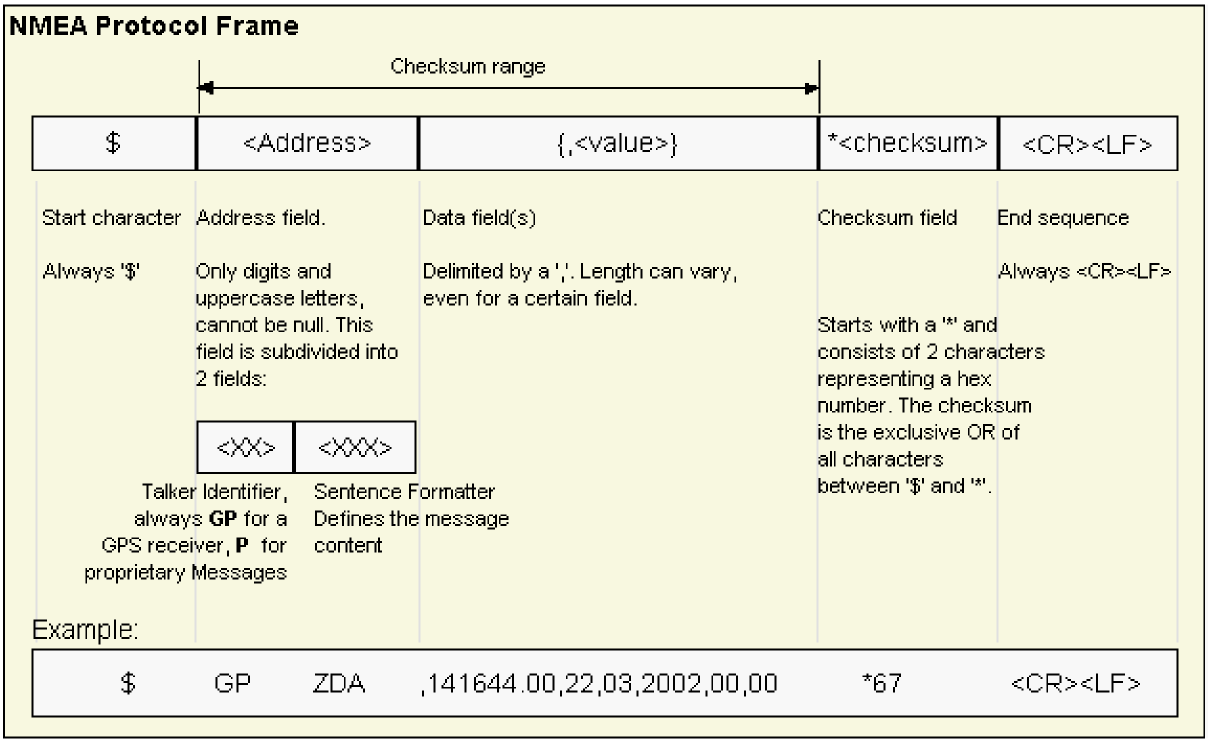
Interface Description

24 NMEA Protocol

24.1 Protocol Overview

24.1.1 Message Format

NMEA messages sent by the GNSS receiver are based on NMEA 0183 Version 4.1. The following picture shows the structure of a NMEA protocol message.



For further information on the NMEA Standard, refer to NMEA 0183 Standard For Interfacing Marine Electronic Devices, Version 4.10, June, 2012. See http://www.nmea.org/ for ordering instructions.

The NMEA standard allows for proprietary, manufacturer-specific messages to be added. These shall be marked with a manufacturer mnemonic. The mnemonic assigned to u-blox is UBX and is used for all non-standard messages. These proprietary NMEA messages therefore have the address field set to PUBX. The first data field in a PUBX message identifies the message number with two digits.

24.1.2 Talker ID

One of the ways the NMEA standard differentiates between GNSS is by using a two-letter message identifier, the 'Talker ID'. The specific Talker ID used by a u-blox receiver will depend on the device model and system configuration. The table below shows the Talker ID that will be used for various GNSS configurations.

NMEA Talker IDs

Configured GNSS	Talker ID
GPS, SBAS, QZSS	GP
GLONASS	GL
Galileo	GA
BeiDou	GB
Any combination of GNSS	GN

24.1.3 Protocol Configuration

The NMEA protocol on u-blox receivers can be configured to the need of customer applications using UBX-CFG-NMEA. For backwards compatibility various versions of this message are supported, however, any new users should use the version that is not marked as deprecated.

There are four NMEA standards supported. The default NMEA version is 4.10. Alternatively versions 4.00, 2.3, and 2.1 can be enabled (for details on how this affects the output refer to section Position Fix Flags in NMEA Mode).

☞ Customers using BeiDou and/or Galileo are recommended to select NMEA version 4.1, as earlier versions have no support for these two GNSS.

☞ Customers using High Precision GNSS (HPG) products are recommended to select NMEA version 4.1, as earlier versions do no support the Float RTK (F) and Real Time Kinematic (R) mode indicator flags in all messages.

NMEA defines satellite numbering systems for some, but not all GNSS (this is partly dependent on the NMEA version). Satellite numbers for unsupported GNSS can be configured using UBX-CFG-NMEA. Unknown satellite numbers are always reported as a null NMEA field (i.e. an empty string)

The NMEA specification indicates that the GGA message is GPS specific. However, u-blox receivers support the output of a GGA message for each of the Talker IDs.

NMEA filtering flags

Parameter	Description
Position filtering	Enable to permit positions from failed or invalid fixes to be reported (with the "V" status flag to indicate that the data is not valid).
Valid position filtering	Enable to permit positions from invalid fixes to be reported (with the "V" status flag to indicate that the data is not valid).
Time filtering	Enable to permit the receiver's best knowledge of time to be output, even though it might be wrong.
Date filtering	Enable to permit the receiver's best knowledge of date to be output, even though it might be wrong.
GPS-only filtering	Enable to restrict output to only report GPS satellites.
Track filtering	Enable to permit course over ground (COG) to be reported even when it would otherwise be frozen.

NMEA flags

Parameter	Description

NMEA flags continued

Parameter	Description
Compatibility Mode	Some older NMEA applications expect the NMEA output to be formatted in a specific way, for example, they will only work if the latitude and longitude have exactly four digits behind the decimal point. u-blox receivers offer a compatibility mode to support these legacy applications.
Consideration Mode	u-blox receivers use a sophisticated signal quality detection scheme, in order to produce the best possible position output. This algorithm considers all SV measurements, and may eventually decide to only use a subset thereof, if it improves the overall position accuracy. If Consideration mode is enabled, all satellites, which were considered for navigation, are communicated as being used for the position determination. If Consideration Mode is disabled, only those satellites which after the consideration step remained in the position output are marked as being used.
Limit82 Mode	Enabling this mode will limit the NMEA sentence length to a maximum of 82 characters.
High Precision Mode	Enabling this mode increases precision of the position output. Latitude and longitude then have seven digits after the decimal point, and altitude has three digits after the decimal point. Note: The High Precision Mode cannot be set in conjunction with either Compatibility Mode or Limit82 Mode.

Extended configuration

Option	Description
GNSS to filter	Filters satellites based on their GNSS
Satellite numbering	This field configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID.
Main Talker ID	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see `UBX-CFG-GNSS`). This field enables the main Talker ID to be overridden.
GSV Talker ID	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden.
BDS Talker ID	By default the Talker ID for BeiDou is 'GB'. This field enables the BeiDou Talker ID to be overridden.

Extra fields in NMEA 4.1 and above

Message	Extra fields
GBS	systemId, signalId
GNS	navStatus
GRS	systemId, signalId
GSA	systemId
GSV	signalId
RMC	navStatus

24.1.4 Satellite Numbering

The NMEA protocol (V4.1) identifies GNSS satellites with a one digit system ID and a two digit satellite number. u-blox receivers support this method in their NMEA output when "strict" SV numbering is selected. In most cases this is the default setting, but can be checked or set using

UBX-CFG-NMEA.

In order to support QZSS within current receivers and prepare for support of other systems (e.g. Galileo) in future receivers, an "extended" SV numbering scheme can be enabled (using UBX-CFG-NMEA). This uses the NMEA-defined numbers where possible, but adds other number ranges to support other GNSS. Note however that these non-standard extensions require 3 digit numbers, which may not be supported by some NMEA parsing software. For example QZSS satellites are reported using numbers in the range 193 to 197.

See Satellite Numbering for a complete list of satellite numbers.

GLONASS satellites can be tracked before they have been identified. In NMEA output, such unknown satellite numbers are always reported as a null field (i.e. an empty string).

24.1.5 Latitude and Longitude Format

According to the NMEA Standard, Latitude and Longitude are output in the format Degrees, Minutes and (Decimal) Fractions of Minutes. To convert to Degrees and Fractions of Degrees, or Degrees, Minutes, Seconds and Fractions of seconds, the 'Minutes' and 'Fractional Minutes' parts need to be converted. In other words: If the GPS Receiver reports a Latitude of 4717.112671 North and Longitude of 00833.914843 East, this is

Latitude 47 Degrees, 17.112671 Minutes

Longitude 8 Degrees, 33.914843 Minutes

or

Latitude 47 Degrees, 17 Minutes, 6.76026 Seconds

Longitude 8 Degrees, 33 Minutes, 54.89058 Seconds

or

Latitude 47.28521118 Degrees

Longitude 8.56524738 Degrees

24.1.6 Position Fix Flags

This section shows how u-blox implements the NMEA protocol and the conditions determining how flags are set.

Flags in NMEA 4.1 and above

NMEA Message Field	GLL, RMC status	GGA quality	GLL, VTG posMode	RMC, GNS posMode
No position fix (at power-up, after losing satellite lock)	V	0	N	N
GNSS fix, but user limits exceeded	V	0	N	N
Dead reckoning fix, but user limits exceeded	V	6	E	E
Dead reckoning fix	A	6	E	E
RTK float	A	5	D	F
RTK fixed	A	4	D	R
2D GNSS fix	A	1 / 2	A / D	A / D
3D GNSS fix	A	1 / 2	A / D	A / D
Combined GNSS/dead reckoning fix	A	1 / 2	A / D	A / D
	See below (1)	See below (2)	See below (3)	See below (3)

(1) Possible values for status: V = Data invalid, A = Data valid

(2) Possible values for quality: 0 = No fix, 1 = Autonomous GNSS fix, 2 = Differential GNSS fix, 4 = RTK fixed, 5 = RTK float, 6 = Estimated/Dead reckoning fix

(3) Possible values for posMode: N = No fix, E = Estimated/Dead reckoning fix, A = Autonomous GNSS fix, D = Differential GNSS fix, F = RTK float, R = RTK fixed

Flags in NMEA 2.3 and above

NMEA Message Field	GLL, RMC status	GGA quality	GSA navMode	GLL, VTG, RMC, GNS posMode
No position fix (at power-up, after losing satellite lock)	V	0	1	N
GNSS fix, but user limits exceeded	V	0	1	N
Dead reckoning fix, but user limits exceeded	V	6	2	E
Dead reckoning fix	A	6	2	E
2D GNSS fix	A	1 / 2	2	A / D
3D GNSS fix	A	1 / 2	3	A / D
Combined GNSS/dead reckoning fix	A	1 / 2	3	A / D
	See below (1)	See below (2)	See below (3)	See below (4)

(1) Possible values for status: V = Data invalid, A = Data valid

(2) Possible values for quality: 0 = No fix, 1 = Autonomous GNSS fix, 2 = Differential GNSS fix, 4 = RTK fixed, 5 = RTK float, 6 = Estimated/Dead reckoning fix

(3) Possible values for navMode: 1 = No fix, 2 = 2D fix, 3 = 3D fix

(4) Possible values for posMode: N = No fix, E = Estimated/Dead reckoning fix, A = Autonomous GNSS fix, D = Differential GNSS fix, F = RTK float, R = RTK fixed

Flags in NMEA 2.1 and below

The flags in NMEA 2.1 and below are the same as NMEA 2.3 and above but with the following differences:

- The posMode field is not output for GLL, RMC and VTG messages (each message has one field less).
- The GGA quality field is set to 1 (instead of 6) for both types of dead reckoning fix.

24.1.7 Multi-GNSS Considerations

Many applications which process NMEA messages assume that only a single GNSS is active. However, when multiple GNSS are configured, the NMEA specification requires the output to change in the following ways:

NMEA output for Multi-GNSS

Change	Description
Main Talker ID	The main Talker ID will be 'GN' (e.g. instead of 'GP' for a GPS receiver)
GSV Talker IDs	The GSV message reports the signal strength of the visible satellites. However, the Talker ID it uses is specific to the GNSS it is reporting information for, so for a multi-GNSS receiver it will not be the same as the main Talker ID. (e.g. other messages will be using the 'GN' Talker ID but the GSV message will use GNSS-specific Talker IDs)

NMEA output for Multi-GNSS continued

Change	Description
Multiple GSA and GRS Messages	Multiple GSA and GRS messages are output for each fix, one for each GNSS. This may confuse applications which assume they are output only once per position fix (as is the case for a single GNSS receiver).

24.1.8 Output of Invalid/Unknown Data

By default the receiver will not output invalid data. In such cases, it will output empty fields.

A valid position fix is reported as follows:

```
$GPGLL,4717.11634,N,00833.91297,E,124923.00,A,A*6E
```

An invalid position fix (but time valid) is reported as follows:

```
$GPGLL,,,,,124924.00,V,N*42
```

If Time is unknown (e.g. during a cold-start):

```
$GPGLL,,,,,,V,N*64
```

Note:

Output of invalid data marked with the 'Invalid/Valid' Flags can be enabled using the UBX protocol message UBX-CFG-NMEA.

24.1.9 Messages Overview

When configuring NMEA messages using the UBX protocol message UBX-CFG-MSG, the Class/Ids shown in the table shall be used.

Page	Mnemonic	Cls/ID	Description
	NMEA Standard Messages		**Standard Messages**
76	DTM	0xF0 0x0A	Datum Reference
77	GBQ	0xF0 0x44	Poll a standard message (if the current Talker ID is GB)
77	GBS	0xF0 0x09	GNSS Satellite Fault Detection
78	GGA	0xF0 0x00	Global positioning system fix data
79	GLL	0xF0 0x01	Latitude and longitude, with time of position fix and status
80	GLQ	0xF0 0x43	Poll a standard message (if the current Talker ID is GL)
81	GNQ	0xF0 0x42	Poll a standard message (if the current Talker ID is GN)
81	GNS	0xF0 0x0D	GNSS fix data
82	GPQ	0xF0 0x40	Poll a standard message (if the current Talker ID is GP)
83	GRS	0xF0 0x06	GNSS Range Residuals
84	GSA	0xF0 0x02	GNSS DOP and Active Satellites
85	GST	0xF0 0x07	GNSS Pseudo Range Error Statistics
86	GSV	0xF0 0x03	GNSS Satellites in View
87	RMC	0xF0 0x04	Recommended Minimum data
88	TXT	0xF0 0x41	Text Transmission
89	VLW	0xF0 0x0F	Dual ground/water distance
89	VTG	0xF0 0x05	Course over ground and Ground speed

NMEA Messages Overview continued

Page	Mnemonic	Cls/ID	Description
90	**ZDA**	0xF0 0x08	Time and Date
	NMEA PUBX Messages		**Proprietary Messages**
92	**CONFIG**	0xF1 0x41	Set Protocols and Baudrate
93	**POSITION**	0xF1 0x00	Lat/Long Position Data
94	**RATE**	0xF1 0x40	Set NMEA message output rate
95	**SVSTATUS**	0xF1 0x03	Satellite Status
96	**TIME**	0xF1 0x04	Time of Day and Clock Information

24.2 Standard Messages

Standard Messages: i.e. Messages as defined in the NMEA Standard.

24.2.1 DTM

24.2.1.1 Datum Reference

Message	**DTM**		
Description	**Datum Reference**		
Type	Output		
Comment	This message gives the difference between the current datum and the reference datum. The current datum defaults to WGS84. The reference datum cannot be changed and is always set to WGS84.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x0A	11	

Message Structure:

```
$xxDTM,datum,subDatum,lat,NS,lon,EW,alt,refDatum*cs<CR><LF>
```

Example:

```
$GPDTM,W84,,0.0,N,0.0,E,0.0,W84*6F

$GPDTM,999,,0.08,N,0.07,E,-47.7,W84*1C
```

Field No.	Name	Unit	Format	Example	Description
0	xxDTM	-	string	$GPDTM	DTM Message ID (xx = current Talker ID)
1	datum	-	string	W84	Local datum code: W84 = WGS84, 999 = user defined
2	subDatum	-	string	-	A null field
3	lat	min	numeric	0.08	Offset in Latitude
4	NS	-	character	S	North/South indicator
5	lon	min	numeric	0.07	Offset in Longitude
6	EW	-	character	E	East/West indicator
7	alt	m	numeric	-2.8	Offset in altitude
8	refDatum	-	string	W84	Reference datum code (always W84 = WGS 84)
9	cs	-	hexadecimal	*67	Checksum
10	<CR><LF>	-	character	-	Carriage return and line feed

24.2.2 GBQ

24.2.2.1 Poll a standard message (if the current Talker ID is GB)

Message	**GBQ**		
Description	**Poll a standard message (if the current Talker ID is GB)**		
Type	Poll Request		
Comment	Polls a standard NMEA message if the current Talker ID is GB		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x44	4	

Message Structure:

```
$xxGBQ,msgId*cs<CR><LF>
```

Example:

```
$EIGBQ,RMC*28
```

Field No.	Name	Unit	Format	Example	Description
0	xxGBQ	-	string	$EIGBQ	GBQ Message ID (xx = Talker ID of the device requesting the poll)
1	msgId	-	string	RMC	Message ID of the message to be polled
2	cs	-	hexadecimal	*28	Checksum
3	<CR><LF>	-	character	-	Carriage return and line feed

24.2.3 GBS

24.2.3.1 GNSS Satellite Fault Detection

Message	**GBS**		
Description	**GNSS Satellite Fault Detection**		
Type	Output		
Comment	This message outputs the results of the Receiver Autonomous Integrity Monitoring Algorithm (RAIM). • The fields **errLat**, **errLon** and **errAlt** output the standard deviation of the position calculation, using all satellites which pass the RAIM test successfully. • The fields **errLat**, **errLon** and **errAlt** are only output if the RAIM process passed successfully (i.e. no or successful edits happened). These fields are never output if 4 or fewer satellites are used for the navigation calculation (because, in such cases, integrity can not be determined by the receiver autonomously). • The fields **prob**, **bias** and **stdev** are only output if at least one satellite failed in the RAIM test. If more than one satellites fail the RAIM test, only the information for the worst satellite is output in this message.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x09	13	

Message Structure:

```
$xxGBS,time,errLat,errLon,errAlt,svid,prob,bias,stddev,systemId,signalId*cs<CR><LF>
```

Example:

```
$GPGBS,235503.00,1.6,1.4,3.2,,,,,,*40
```

```
$GPGBS,235458.00,1.4,1.3,3.1,03,,-21.4,3.8,1,0*5B
```

Field No.	Name	Unit	Format	Example	Description
0	xxGBS	-	string	$GPGBS	GBS Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	235503.00	UTC time to which this RAIM sentence belongs, see note on UTC representation
2	errLat	m	numeric	1.6	Expected error in latitude
3	errLon	m	numeric	1.4	Expected error in longitude
4	errAlt	m	numeric	3.2	Expected error in altitude
5	svid	-	numeric	03	Satellite ID of most likely failed satellite
6	prob	-	numeric	-	Probability of missed detection, not supported (empty)
7	bias	m	numeric	-21.4	Estimate on most likely failed satellite (a priori residual)
8	stddev	m	numeric	3.8	Standard deviation of estimated bias
9	systemId	-	numeric	1	NMEA defined GNSS System ID **NMEA v4.10 and above only**
10	signalId	-	numeric	0	NMEA defined GNSS Signal ID (0 = All signals, see Signal Identifiers table for other values) **NMEA v4.10 and above only**
11	cs	-	hexadecimal	*5B	Checksum
12	<CR><LF>	-	character	-	Carriage return and line feed

24.2.4 GGA

24.2.4.1 Global positioning system fix data

Message	**GGA**		
Description	**Global positioning system fix data**		
Type	Output		
Comment	**The output of this message is dependent on the currently selected datum (default: WGS84). The NMEA specification indicates that the GGA message is GPS specific. However, when the receiver is configured for multi-GNSS, the GGA message contents will be generated from the multi-GNSS solution. For multi-GNSS use, it is recommended that the NMEA-GNS message is used instead.** Time and position, together with GPS fixing related data (number of satellites in use, and the resulting HDOP, age of differential data if in use, etc.).		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x00	17	

Message Structure:

```
$xxGGA,time,lat,NS,long,EW,quality,numSV,HDOP,alt,M,sep,M,diffAge,diffStation*cs<CR><LF>
```

Example:

```
$GPGGA,092725.00,4717.11399,N,00833.91590,E,1,08,1.01,499.6,M,48.0,M,,*5B
```

Field No.	Name	Unit	Format	Example	Description
0	xxGGA	-	string	$GPGGA	GGA Message ID (xx = current Talker ID)

GGA continued

Field No.	Name	Unit	Format	Example	Description
1	time	-	hhmmss.ss	092725.00	UTC time, see note on UTC representation
2	lat	-	ddmm. mmmmm	4717.11399	Latitude (degrees & minutes), see format description
3	NS	-	character	N	North/South indicator
4	long	-	dddmm. mmmmm	00833.91590	Longitude (degrees & minutes), see format description
5	EW	-	character	E	East/West indicator
6	quality	-	digit	1	Quality indicator for position fix: 0 = No Fix / Invalid 1 = Standard GPS (2D/3D) 2 = Differential GPS 4 = RTK fixed solution 5 = RTK float solution 6 = Estimated (DR) Fix See also position fix flags description.
7	numSV	-	numeric	08	Number of satellites used (range: 0-12)
8	HDOP	-	numeric	1.01	Horizontal Dilution of Precision
9	alt	m	numeric	499.6	Altitude above mean sea level
10	uAlt	-	character	M	Altitude units: meters (fixed field)
11	sep	m	numeric	48.0	Geoid separation: difference between ellipsoid and mean sea level
12	uSep	-	character	M	Separation units: meters (fixed field)
13	diffAge	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
14	diffStation	-	numeric	-	ID of station providing differential corrections (blank when DGPS is not used)
15	cs	-	hexadecimal	*5B	Checksum
16	<CR><LF>	-	character	-	Carriage return and line feed

24.2.5 GLL

24.2.5.1 Latitude and longitude, with time of position fix and status

Message	**GLL**		
Description	**Latitude and longitude, with time of position fix and status**		
Type	Output		
Comment	**The output of this message is dependent on the currently selected datum (default: WGS84)** -		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x01	10	

Message Structure:

```
$xxGLL,lat,NS,long,EW,time,status,posMode*cs<CR><LF>
```

Example:

```
$GPGLL,4717.11364,N,00833.91565,E,092321.00,A,A*60
```

GLL continued

Field No.	Name	Unit	Format	Example	Description
Field No.	Name	Unit	Format	Example	Description
0	xxGLL	-	string	$GPGLL	GLL Message ID (xx = current Talker ID)
1	lat	-	ddmm. mmmmm	4717.11364	Latitude (degrees & minutes), see format description
2	NS	-	character	N	North/South indicator
3	long	-	dddmm. mmmmm	00833.91565	Longitude (degrees & minutes), see format description
4	EW	-	character	E	East/West indicator
5	time	-	hhmmss.ss	092321.00	UTC time, see note on UTC representation
6	status	-	character	A	V = Data invalid or receiver warning, A = Data valid. See position fix flags description.
7	posMode	-	character	A	Positioning mode, see position fix flags description. NMEA v2.3 and above only
8	cs	-	hexadecimal	*60	Checksum
9	<CR><LF>	-	character	-	Carriage return and line feed

24.2.6 GLQ

24.2.6.1 Poll a standard message (if the current Talker ID is GL)

Message	**GLQ**		
Description	**Poll a standard message (if the current Talker ID is GL)**		
Type	Poll Request		
Comment	Polls a standard NMEA message if the current Talker ID is GL		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x43	4	

Message Structure:

```
$xxGLQ,msgId*cs<CR><LF>
```

Example:

```
$EIGLQ,RMC*3A
```

Field No.	Name	Unit	Format	Example	Description
0	xxGLQ	-	string	$EIGLQ	GLQ Message ID (xx = Talker ID of the device requesting the poll)
1	msgId	-	string	RMC	Message ID of the message to be polled
2	cs	-	hexadecimal	*3A	Checksum
3	<CR><LF>	-	character	-	Carriage return and line feed

24.2.7 GNQ

24.2.7.1 Poll a standard message (if the current Talker ID is GN)

Message	**GNQ**		
Description	**Poll a standard message (if the current Talker ID is GN)**		
Type	Poll Request		
Comment	Polls a standard NMEA message if the current Talker ID is GN		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x42	4	

Message Structure:

```
$xxGNQ,msgId*cs<CR><LF>
```

Example:

```
$EIGNQ,RMC*3A
```

Field No.	Name	Unit	Format	Example	Description
0	xxGNQ	-	string	$EIGNQ	GNQ Message ID (xx = Talker ID of the device requesting the poll)
1	msgId	-	string	RMC	Message ID of the message to be polled
2	cs	-	hexadecimal	*3A	Checksum
3	<CR><LF>	-	character	-	Carriage return and line feed

24.2.8 GNS

24.2.8.1 GNSS fix data

Message	**GNS**		
Description	**GNSS fix data**		
Type	Output		
Comment	**The output of this message is dependent on the currently selected datum (default: WGS84)** Time and position, together with GNSS fixing related data (number of satellites in use, and the resulting HDOP, age of differential data if in use, etc.).		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x0D	16	

Message Structure:

```
$xxGNS,time,lat,NS,long,EW,posMode,numSV,HDOP,alt,altRef,diffAge,diffStation,navStatus*cs<CR><LF>
```

Example:

```
$GPGNS,091547.00,5114.50897,N,00012.28663,W,AA,10,0.83,111.1,45.6,,,V*71
```

Field No.	Name	Unit	Format	Example	Description
0	xxGNS	-	string	$GPGNS	GNS Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	091547.00	UTC time, see note on UTC representation
2	lat	-	ddmm. mmmmm	5114.50897	Latitude (degrees & minutes), see format description
3	NS	-	character	N	North/South indicator
4	long	-	dddmm. mmmmm	00012.28663	Longitude (degrees & minutes), see format description

GNS continued

Field No.	Name	Unit	Format	Example	Description
5	EW	-	character	E	East/West indicator
6	posMode	-	character	AA	Positioning mode, see position fix flags description. First character for GPS, second character for GLONASS
7	numSV	-	numeric	10	Number of satellites used (range: 0-99)
8	HDOP	-	numeric	0.83	Horizontal Dilution of Precision
9	alt	m	numeric	111.1	Altitude above mean sea level
10	sep	m	numeric	45.6	Geoid separation: difference between ellipsoid and mean sea level
11	diffAge	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
12	diffStation	-	numeric	-	ID of station providing differential corrections (blank when DGPS is not used)
13	navStatus	-	character	V	Navigational status indicator (V = Equipment is not providing navigational status information) **NMEA v4.10 and above only**
14	cs	-	hexadecimal	*71	Checksum
15	<CR><LF>	-	character	-	Carriage return and line feed

24.2.9 GPQ

24.2.9.1 Poll a standard message (if the current Talker ID is GP)

Message	**GPQ**		
Description	**Poll a standard message (if the current Talker ID is GP)**		
Type	Poll Request		
Comment	Polls a standard NMEA message if the current Talker ID is GP		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x40	4	

Message Structure:

```
$xxGPQ,msgId*cs<CR><LF>
```

Example:

```
$EIGPQ,RMC*3A
```

Field No.	Name	Unit	Format	Example	Description
0	xxGPQ	-	string	$EIGPQ	GPQ Message ID (xx = Talker ID of the device requesting the poll)
1	msgId	-	string	RMC	Message ID of the message to be polled
2	cs	-	hexadecimal	*3A	Checksum
3	<CR><LF>	-	character	-	Carriage return and line feed

24.2.10 GRS

24.2.10.1 GNSS Range Residuals

Message	**GRS**		
Description	**GNSS Range Residuals**		
Type	Output		
Comment	**This messages relates to associated GGA and GSA messages.** If less than 12 SVs are available, the remaining fields are output empty. If more than 12 SVs are used, only the residuals of the first 12 SVs are output, in order to remain consistent with the NMEA standard. **In a multi-GNSS system this message will be output multiple times, once for each GNSS.**		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x06	19	

Message Structure:

```
$xxGRS,time, mode {,residual},systemId,signalId*cs<CR><LF>
```

Example:

```
$GPGRS,082632.00,1,0.54,0.83,1.00,1.02,-2.12,2.64,-0.71,-1.18,0.25,,,1,0*70
```

Field No.	Name	Unit	Format	Example	Description
0	xxGRS	-	string	$GPGRS	GRS Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	082632.00	UTC time of associated position fix, see note on UTC representation
2	mode	-	digit	1	Mode (u-blox receivers will always output Mode 1 residuals): 0 = Residuals were used to calculate the position given in the matching GGA sentence. 1 = Residuals were recomputed after the GGA position was computed.
Start of repeated block (12 times)					
3 + 1*N	residual	m	numeric	0.54	Range residuals for SVs used in navigation. The SV order matches the order from the GSA sentence.
End of repeated block					
15	systemId	-	numeric	1	NMEA defined GNSS System ID **NMEA v4.10 and above only**
16	signalId	-	numeric	0	NMEA defined GNSS Signal ID (0 = All signals, see Signal Identifiers table for other values) **NMEA v4.10 and above only**
17	cs	-	hexadecimal	*70	Checksum
18	<CR><LF>	-	character	-	Carriage return and line feed

24.2.11 GSA

24.2.11.1 GNSS DOP and Active Satellites

Message	**GSA**		
Description	**GNSS DOP and Active Satellites**		
Type	Output		
Comment	The GNSS receiver operating mode, satellites used for navigation, and DOP values. • If less than 12 SVs are used for navigation, the remaining fields are left empty. If more than 12 SVs are used for navigation, only the IDs of the first 12 are output. • The SV numbers (fields 'sv') are in the range of 1 to 32 for GPS satellites, and 33 to 64 for SBAS satellites (33 = SBAS PRN 120, 34 = SBAS PRN 121, and so on) **In a multi-GNSS system this message will be output multiple times, once for each GNSS.**		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x02	21	

Message Structure:

```
$xxGSA,opMode,navMode{,sv},PDOP,HDOP,VDOP,systemId*cs<CR><LF>
```

Example:

```
$GPGSA,A,3,23,29,07,08,09,18,26,28,,,,,1.94,1.18,1.54,1*0D
```

Field No.	Name	Unit	Format	Example	Description
0	xxGSA	-	string	$GPGSA	GSA Message ID (xx = current Talker ID)
1	opMode	-	character	A	Operation mode: M = Manually set to operate in 2D or 3D mode A = Automatically switching between 2D or 3D mode
2	navMode	-	digit	3	Navigation mode (see also position fix flags description): 1 = Fix not available 2 = 2D Fix 3 = 3D Fix
Start of repeated block (12 times)					
3 + 1*N	sv	-	numeric	29	Satellite number
End of repeated block					
15	PDOP	-	numeric	1.94	Position dilution of precision
16	HDOP	-	numeric	1.18	Horizontal dilution of precision
17	VDOP	-	numeric	1.54	Vertical dilution of precision
18	systemId	-	numeric	1	NMEA defined GNSS System ID **NMEA v4.10 and above only**
19	cs	-	hexadecimal	*0D	Checksum
20	<CR><LF>	-	character	-	Carriage return and line feed

24.2.12 GST

24.2.12.1 GNSS Pseudo Range Error Statistics

Message	**GST**		
Description	**GNSS Pseudo Range Error Statistics**		
Type	Output		
Comment	This message reports statistical information on the quality of the position solution.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x07	11	

Message Structure:

```
$xxGST,time,rangeRms,stdMajor,stdMinor,orient,stdLat,stdLong,stdAlt*cs<CR><LF>
```

Example:

```
$GPGST,082356.00,1.8,,,,1.7,1.3,2.2*7E
```

Field No.	Name	Unit	Format	Example	Description
0	xxGST	-	string	$GPGST	GST Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	082356.00	UTC time of associated position fix, see note on UTC representation
2	rangeRms	m	numeric	1.8	RMS value of the standard deviation of the ranges
3	stdMajor	m	numeric	-	Standard deviation of semi-major axis (only supported in ADR 4.10 and above)
4	stdMinor	m	numeric	-	Standard deviation of semi-minor axis (only supported in ADR 4.10 and above)
5	orient	deg	numeric	-	Orientation of semi-major axis (only supported in ADR 4.10 and above)
6	stdLat	m	numeric	1.7	Standard deviation of latitude error
7	stdLong	m	numeric	1.3	Standard deviation of longitude error
8	stdAlt	m	numeric	2.2	Standard deviation of altitude error
9	cs	-	hexadecimal	*7E	Checksum
10	<CR><LF>	-	character	-	Carriage return and line feed

24.2.13 GSV

24.2.13.1 GNSS Satellites in View

Message	**GSV**		
Description	**GNSS Satellites in View**		
Type	Output		
Comment	The number of satellites in view, together with each SV ID, elevation azimuth, and signal strength (C/No) value. Only four satellite details are transmitted in one message. **In a multi-GNSS system sets of GSV messages will be output multiple times, one set for each GNSS.**		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x03	8..16	

Message Structure:

```
$xxGSV,numMsg,msgNum,numSV,{,sv,elv,az,cno},signalId*cs<CR><LF>
```

Example:

```
$GPGSV,3,1,10,23,38,230,44,29,71,156,47,07,29,116,41,08,09,081,36,0*7F
$GPGSV,3,2,10,10,07,189,,05,05,220,,09,34,274,42,18,25,309,44,0*72
$GPGSV,3,3,10,26,82,187,47,28,43,056,46,0*77
```

Field No.	Name	Unit	Format	Example	Description
0	xxGSV	-	string	$GPGSV	GSV Message ID (xx = GSV Talker ID)
1	numMsg	-	digit	3	Number of messages, total number of GSV messages being output
2	msgNum	-	digit	1	Number of this message
3	numSV	-	numeric	10	Number of satellites in view
Start of repeated block (1..4 times)					
4 + 4*N	sv	-	numeric	23	Satellite ID
5 + 4*N	elv	deg	numeric	38	Elevation (range 0-90)
6 + 4*N	az	deg	numeric	230	Azimuth, (range 0-359)
7 + 4*N	cno	dB Hz	numeric	44	Signal strength (C/N0, range 0-99), blank when not tracking
End of repeated block					
5.. 16	signalId	-	numeric	0	NMEA defined GNSS Signal ID (0 = All signals, see Signal Identifiers table for other values) **NMEA v4.10 and above only**
6.. 16	cs	-	hexadecimal	*7F	Checksum
7.. 16	<CR><LF>	-	character	-	Carriage return and line feed

24.2.14 RMC

24.2.14.1 Recommended Minimum data

Message	**RMC**		
Description	**Recommended Minimum data**		
Type	Output		
Comment	**The output of this message is dependent on the currently selected datum (default: WGS84)** The recommended minimum sentence defined by NMEA for GNSS system data.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x04	16	

Message Structure:

```
$xxRMC,time,status,lat,NS,long,EW,spd,cog,date,mv,mvEW,posMode,navStatus*cs<CR><LF>
```

Example:

```
$GPRMC,083559.00,A,4717.11437,N,00833.91522,E,0.004,77.52,091202,,,A,V*57
```

Field No.	Name	Unit	Format	Example	Description
0	xxRMC	-	string	$GPRMC	RMC Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	083559.00	UTC time, see note on UTC representation
2	status	-	character	A	Status, V = Navigation receiver warning, A = Data valid, see position fix flags description
3	lat	-	ddmm.mmmmm	4717.11437	Latitude (degrees & minutes), see format description
4	NS	-	character	N	North/South indicator
5	long	-	dddmm.mmmmm	00833.91522	Longitude (degrees & minutes), see format description
6	EW	-	character	E	East/West indicator
7	spd	knots	numeric	0.004	Speed over ground
8	cog	degrees	numeric	77.52	Course over ground
9	date	-	ddmmyy	091202	Date in day, month, year format, see note on UTC representation
10	mv	degrees	numeric	-	Magnetic variation value. Only supported in ADR 4.10 and above.
11	mvEW	-	character	-	Magnetic variation E/W indicator. Only supported in ADR 4.10 and above.
12	posMode	-	character	A	Mode Indicator, see position fix flags description **NMEA v2.3 and above only**
13	navStatus	-	character	V	Navigational status indicator (V = Equipment is not providing navigational status information) **NMEA v4.10 and above only**

RMC continued

Field No.	Name	Unit	Format	Example	Description
14	cs	-	hexadecimal	*57	Checksum
15	<CR><LF>	-	character	-	Carriage return and line feed

24.2.15 TXT

24.2.15.1 Text Transmission

Message	**TXT**		
Description	**Text Transmission**		
Type	Output		
Comment	**This message is not configured through UBX-CFG-MSG, but instead through UBX-CFG-INF.** This message outputs various information on the receiver, such as power-up screen, software version etc. This message can be configured using UBX Protocol message UBX-CFG-INF.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x41	7	

Message Structure:

```
$xxTXT,numMsg,msgNum,msgType,text*cs<CR><LF>
```

Example:

```
$GPTXT,01,01,02,u-blox ag - www.u-blox.com*50
$GPTXT,01,01,02,ANTARIS ATR0620 HW 00000040*67
```

Field No.	Name	Unit	Format	Example	Description
0	xxTXT	-	string	$GPTXT	TXT Message ID (xx = current Talker ID)
1	numMsg	-	numeric	01	Total number of messages in this transmission, 01..99
2	msgNum	-	numeric	01	Message number in this transmission, range 01..xx
3	msgType	-	numeric	02	Text identifier, u-blox receivers specify the type of the message with this number. 00: Error 01: Warning 02: Notice 07: User
4	text	-	string	www.u-blox.com	Any ASCII text
5	cs	-	hexadecimal	*67	Checksum
6	<CR><LF>	-	character	-	Carriage return and line feed

24.2.16 VLW

24.2.16.1 Dual ground/water distance

Message	**VLW**		
Description	**Dual ground/water distance**		
Type	Output		
Comment	The distance traveled, relative to the water and over the ground. This message relates to the Odometer functionality.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x0F	11	

Message Structure:

```
$xxVLW,twd,twdUnit,wd,wdUnit,tgd,tgdUnit,gd,gdUnit*cs<CR><LF>
```

Example:

```
$GPVLW,,N,,N,15.8,N,1.2,N*06
```

Field No.	Name	Unit	Format	Example	Description
0	xxVLW	-	string	$GPVLW	VLW Message ID (xx = current Talker ID)
1	twd	nm	numeric	-	Total cumulative water distance, not output
2	twdUnit	-	character	N	Fixed field: nautical miles
3	wd	nm	numeric	-	Water distance since reset, not output
4	wdUnit	-	character	N	Fixed field: nautical miles
5	tgd	nm	numeric	15.8	Total cumulative ground distance
6	tgdUnit	-	character	N	Fixed field: nautical miles
7	gd	nm	numeric	1.2	Ground distance since reset
8	gdUnit	-	character	N	Fixed field: nautical miles
9	cs	-	hexadecimal	*06	Checksum
10	<CR><LF>	-	character	-	Carriage return and line feed

24.2.17 VTG

24.2.17.1 Course over ground and Ground speed

Message	**VTG**		
Description	**Course over ground and Ground speed**		
Type	Output		
Comment	Velocity is given as Course over Ground (COG) and Speed over Ground (SOG).		
	ID for CFG-MSG	Number of fields	
Message Info	0xF0 0x05	12	

Message Structure:

```
$xxVTG,cogt,T,cogm,M,knots,N,kph,K,posMode*cs<CR><LF>
```

Example:

```
$GPVTG,77.52,T,,M,0.004,N,0.008,K,A*06
```

Field No.	Name	Unit	Format	Example	Description
0	xxVTG	-	string	$GPVTG	VTG Message ID (xx = current Talker ID)

VTG continued

Field No.	Name	Unit	Format	Example	Description
1	cogt	degrees	numeric	77.52	Course over ground (true)
2	T	-	character	T	Fixed field: true
3	cogm	degrees	numeric	-	Course over ground (magnetic). Only supported in ADR 4.10 and above.
4	M	-	character	M	Fixed field: magnetic
5	knots	knots	numeric	0.004	Speed over ground
6	N	-	character	N	Fixed field: knots
7	kph	km/h	numeric	0.008	Speed over ground
8	K	-	character	K	Fixed field: kilometers per hour
9	posMode	-	character	A	Mode Indicator, see position fix flags description **NMEA v2.3 and above only**
10	cs	-	hexadecimal	*06	Checksum
11	<CR><LF>	-	character	-	Carriage return and line feed

24.2.18 ZDA

24.2.18.1 Time and Date

Message	**ZDA**		
Description	**Time and Date**		
Type	Output		
Comment	-		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x08	9	

Message Structure:

```
$xxZDA,hhmmss.ss,day,month,year,ltzh,ltzn*cs<CR><LF>
```

Example:

```
$GPZDA,082710.00,16,09,2002,00,00*64
```

Field No.	Name	Unit	Format	Example	Description
0	xxZDA	-	string	$GPZDA	ZDA Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	082710.00	UTC Time, see note on UTC representation
2	day	day	dd	16	UTC day (range: 1-31)
3	month	month	mm	09	UTC month (range: 1-12)
4	year	year	yyyy	2002	UTC year
5	ltzh	-	xx	00	Local time zone hours (fixed to 00)
6	ltzn	-	zz	00	Local time zone minutes (fixed to 00)

ZDA continued

Field No.	Name	Unit	Format	Example	Description
7	cs	-	hexadecimal	*64	Checksum
8	<CR><LF>	-	character	-	Carriage return and line feed

24.3 PUBX Messages

Proprietary Messages: i.e. Messages defined by u-blox.

24.3.1 CONFIG (PUBX,41)

24.3.1.1 Set Protocols and Baudrate

Message	**CONFIG**		
Description	**Set Protocols and Baudrate**		
Type	Set		
Comment	-		
Message Info	ID for CFG-MSG	Number of fields	
	0xF1 0x41	9	

Message Structure:

```
$PUBX,41,portId,inProto,outProto,baudrate,autobauding*cs<CR><LF>
```

Example:

```
$PUBX,41,1,0007,0003,19200,0*25
```

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	`msgId`	-	numeric	41	Proprietary message identifier
2	`portId`	-	numeric	1	ID of communication port. For a list of port IDs see Serial Communication Ports Description.
3	`inProto`	-	hexadecimal	0007	Input protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in `UBX-CFG-PRT`.
4	`outProto`	-	hexadecimal	0003	Output protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in `UBX-CFG-PRT`.
5	`baudrate`	bits/s	numeric	19200	Baudrate
6	`autobauding`	-	numeric	0	Autobauding: 1=enable, 0=disable (not supported on u-blox 5, set to 0)
7	`cs`	-	hexadecimal	*25	Checksum
8	`<CR><LF>`	-	character	-	Carriage return and line feed

24.3.2 POSITION (PUBX,00)

24.3.2.1 Lat/Long Position Data

Message	**POSITION**		
Description	**Lat/Long Position Data**		
Type	Output		
Comment	**The output of this message is dependent on the currently selected datum (default: WGS84)** This message contains position solution data. The datum selection may be changed using the message UBX-CFG-DAT.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF1 0x00	23	

Message Structure:

```
$PUBX,00,time,lat,NS,long,EW,altRef,navStat,hAcc,vAcc,SOG,COG,vVel,diffAge,HDOP,VDOP,TDOP,numSvs,reserved,DR,*cs<CR><LF>
```

Example:

```
$PUBX,00,081350.00,4717.113210,N,00833.915187,E,546.589,G3,2.1,2.0,0.007,77.52,0.007,,0.92,1.19,0.77,9,0,0*5F
```

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	00	Proprietary message identifier: 00
2	time	-	hhmmss.ss	081350.00	UTC time, see note on UTC representation
3	lat	-	ddmm.mmmmm	4717.113210	Latitude (degrees & minutes), see format description
4	NS	-	character	N	North/South Indicator
5	long	-	dddmm.mmmmm	00833.915187	Longitude (degrees & minutes), see format description
6	EW	-	character	E	East/West indicator
7	altRef	m	numeric	546.589	Altitude above user datum ellipsoid.
8	navStat	-	string	G3	Navigation Status: NF = No Fix DR = Dead reckoning only solution G2 = Stand alone 2D solution G3 = Stand alone 3D solution D2 = Differential 2D solution D3 = Differential 3D solution RK = Combined GPS + dead reckoning solution TT = Time only solution
9	hAcc	m	numeric	2.1	Horizontal accuracy estimate.
10	vAcc	m	numeric	2.0	Vertical accuracy estimate.
11	SOG	km/h	numeric	0.007	Speed over ground
12	COG	deg	numeric	77.52	Course over ground

POSITION continued

Field No.	Name	Unit	Format	Example	Description
13	vVel	m/s	numeric	0.007	Vertical velocity (positive downwards)
14	diffAge	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
15	HDOP	-	numeric	0.92	HDOP, Horizontal Dilution of Precision
16	VDOP	-	numeric	1.19	VDOP, Vertical Dilution of Precision
17	TDOP	-	numeric	0.77	TDOP, Time Dilution of Precision
18	numSvs	-	numeric	9	Number of satellites used in the navigation solution
19	reserved	-	numeric	0	Reserved, always set to 0
20	DR	-	numeric	0	DR used
21	cs	-	hexadecimal	*5B	Checksum
22	<CR><LF>	-	character	-	Carriage return and line feed

24.3.3 RATE (PUBX,40)

24.3.3.1 Set NMEA message output rate

Message	**RATE**		
Description	**Set NMEA message output rate**		
Type	Set		
Comment	Set/Get message rate configuration (s) to/from the receiver. • Send rate is relative to the event a message is registered on. For example, if the rate of a navigation message is set to 2, the message is sent every second navigation solution.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF1 0x40	11	

Message Structure:

```
$PUBX,40,msgId,rddc,rus1,rus2,rusb,rspi,reserved*cs<CR><LF>
```

Example:

```
$PUBX,40,GLL,1,0,0,0,0,0*5D
```

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	ID	-	numeric	40	Proprietary message identifier
2	msgId	-	string	GLL	NMEA message identifier
3	rddc	cycles	numeric	1	output rate on DDC 0 disables that message from being output on this port 1 means that this message is output every epoch

RATE continued

Field No.	Name	Unit	Format	Example	Description
4	rus1	cycles	numeric	1	output rate on USART 1 0 disables that message from being output on this port 1 means that this message is output every epoch
5	rus2	cycles	numeric	1	output rate on USART 2 0 disables that message from being output on this port 1 means that this message is output every epoch
6	rusb	cycles	numeric	1	output rate on USB 0 disables that message from being output on this port 1 means that this message is output every epoch
7	rspi	cycles	numeric	1	output rate on SPI 0 disables that message from being output on this port 1 means that this message is output every epoch
8	reserved	-	numeric	0	Reserved: always fill with 0
9	cs	-	hexadecimal	*5D	Checksum
10	<CR><LF>	-	character	-	Carriage return and line feed

24.3.4 SVSTATUS (PUBX,03)

24.3.4.1 Satellite Status

Message	**SVSTATUS**		
Description	**Satellite Status**		
Type	Output		
Comment	The PUBX,03 message contains satellite status information.		
	ID for CFG-MSG	Number of fields	
Message Info	0xF1 0x03	5 + 6*n	

Message Structure:

```
$PUBX,03,GT{,sv,s,az,el,cno,lck},*cs<CR><LF>
```

Example:

```
$PUBX,03,11,23,-,,,45,010,29,-,,,46,013,07,-,,,42,015,08,U,067,31,42,025,10,U,195,33,46,026,18,U,32
6,08,39,026,17,-,,,32,015,26,U,306,66,48,025,27,U,073,10,36,026,28,U,089,61,46,024,15,-,,,39,014*0D
```

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	03	Proprietary message identifier: 03

SVSTATUS continued

Field No.	Name	Unit	Format	Example	Description
2	n	-	numeric	11	Number of GNSS satellites tracked
Start of repeated block (n times)					
3 + 6*N	sv	-	numeric	23	Satellite ID according to UBX svId mapping (see Satellite Numbering)
4 + 6*N	s	-	character	-	Satellite status: - = Not used U = Used in solution e = Ephemeris available, but not used for navigation
5 + 6*N	az	deg	numeric	-	Satellite azimuth (range: 0-359)
6 + 6*N	el	deg	numeric	-	Satellite elevation (range: 0-90)
7 + 6*N	cno	dB Hz	numeric	45	Signal strength (C/N0, range 0-99), blank when not tracking
8 + 6*N	lck	s	numeric	010	Satellite carrier lock time (range: 0-64) 0: code lock only 64: lock for 64 seconds or more
End of repeated block					
3 + 6*n	cs	-	hexadecimal	*0D	Checksum
4 + 6*n	<CR><LF>	-	character	-	Carriage return and line feed

24.3.5 TIME (PUBX,04)

24.3.5.1 Time of Day and Clock Information

Message	**TIME**		
Description	**Time of Day and Clock Information**		
Type	Output		
Comment	-		
	ID for CFG-MSG	Number of fields	
Message Info	0xF1 0x04	12	

Message Structure:

```
$PUBX,04,time,date,utcTow,utcWk,leapSec,clkBias,clkDrift,tpGran,*cs<CR><LF>
```

Example:

```
$PUBX,04,073731.00,091202,113851.00,1196,15D,1930035,-2660.664,43,*3C
```

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	04	Proprietary message identifier: 04
2	time	-	hhmmss.ss	073731.00	UTC time, see note on UTC representation

TIME continued

Field No.	Name	Unit	Format	Example	Description
3	date	-	ddmmyy	091202	UTC date, day, month, year format, see note on UTC representation
4	utcTow	s	numeric	113851.00	UTC Time of Week
5	utcWk	-	numeric	1196	UTC week number, continues beyond 1023
6	leapSec	s	numeric/text	15D	Leap seconds The number is marked with a ‘D’ if the value is the firmware default value. If the value is not marked it has been received from a satellite.
7	clkBias	ns	numeric	1930035	Receiver clock bias
8	clkDrift	ns/s	numeric	-2660.664	Receiver clock drift
9	tpGran	ns	numeric	43	Time Pulse Granularity, The quantization error of the TIMEPULSE pin
10	cs	-	hexadecimal	*3C	Checksum
11	<CR><LF>	-	character	-	Carriage Return and Line Feed

25 UBX Protocol

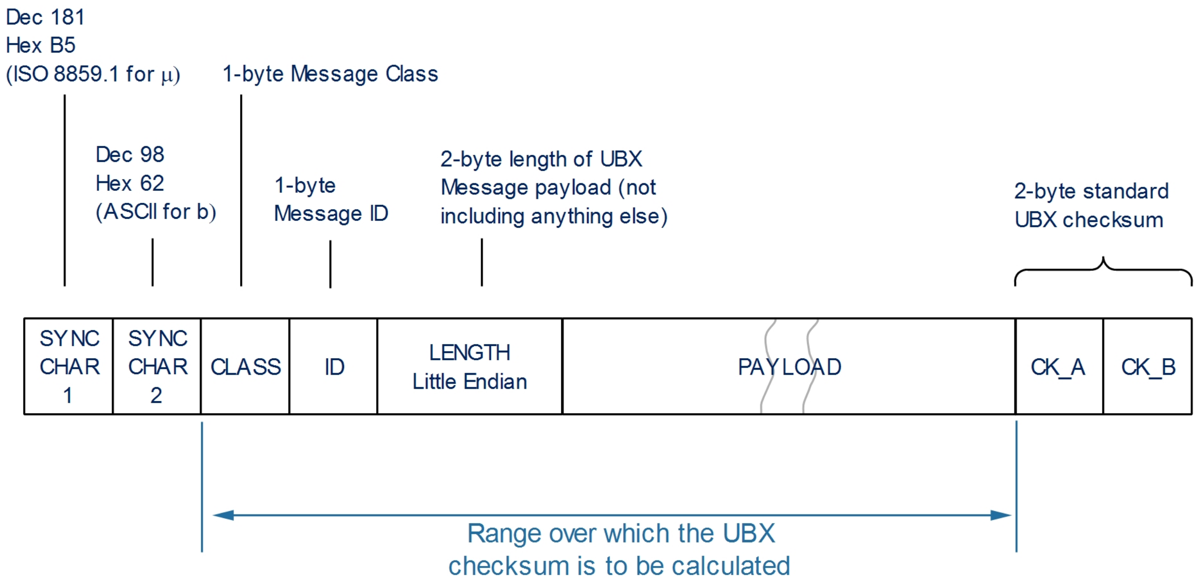
25.1 UBX Protocol Key Features

u-blox receivers support a u-blox proprietary protocol to communicate with a host computer. This protocol has the following key features:

- Compact - uses 8-bit Binary Data.
- Checksum Protected - uses a low-overhead checksum algorithm
- Modular - uses a 2-stage message identifier (Class and Message ID)

25.2 UBX Frame Structure

The structure of a basic UBX Frame is shown in the following diagram.



- Every **Frame** starts with a 2-byte Preamble consisting of two synchronization characters: 0xB5 0x62.
- A 1-byte Message **Class** field follows. A Class is a group of messages that are related to each other.
- A 1-byte Message **ID** field defines the message that is to follow.
- A 2-byte **Length** field follows. The length is defined as being that of the payload only. It does not include the Preamble, Message Class, Message ID, Length, or CRC fields. The number format of the length field is a Little-Endian unsigned 16-bit integer.
- The **Payload** field contains a variable number of bytes.
- The two 1-byte **CK_A** and **CK_B** fields hold a 16-bit checksum whose calculation is defined below. This concludes the Frame.

25.3 UBX Payload Definition Rules

25.3.1 Structure Packing

Values are placed in an order that structure packing is not a problem. This means that 2-byte values shall start on offsets which are a multiple of 2; 4-byte values shall start at a multiple of 4; and so on.

25.3.2 Reserved Elements

Some messages contain reserved fields or bits to allow for future expansion. The contents of these elements should be ignored in output messages and must be set to zero in input messages. Where a message is output and subsequently returned to the receiver as input message, reserved elements can either be explicitly set to zero or left with whatever value they were output with.

25.3.3 Undefined Values

The description of some fields provide specific meanings for specific values. For example, the field gnssId appears in many UBX messages and uses 0 to indicate GPS, 1 for SBAS and so on (see Satellite Numbering for details); however it is usually stored in a byte with far more possible values than the handful currently defined. All such undefined values are reserved for future expansion and therefore should not be used.

25.3.4 Message Naming

Referring to messages is done by adding the class name and a dash in front of the message name. For example, the version information message is referred to as `UBX-MON-VER`. Referring to message fields or their values is done by adding a dot and the name, e.g. `UBX-MON-VER.swVersion`.

25.3.5 Number Formats

All multi-byte values are ordered in Little Endian format, unless otherwise indicated.

All floating point values are transmitted in IEEE754 single or double precision.

Variable Type Definitions

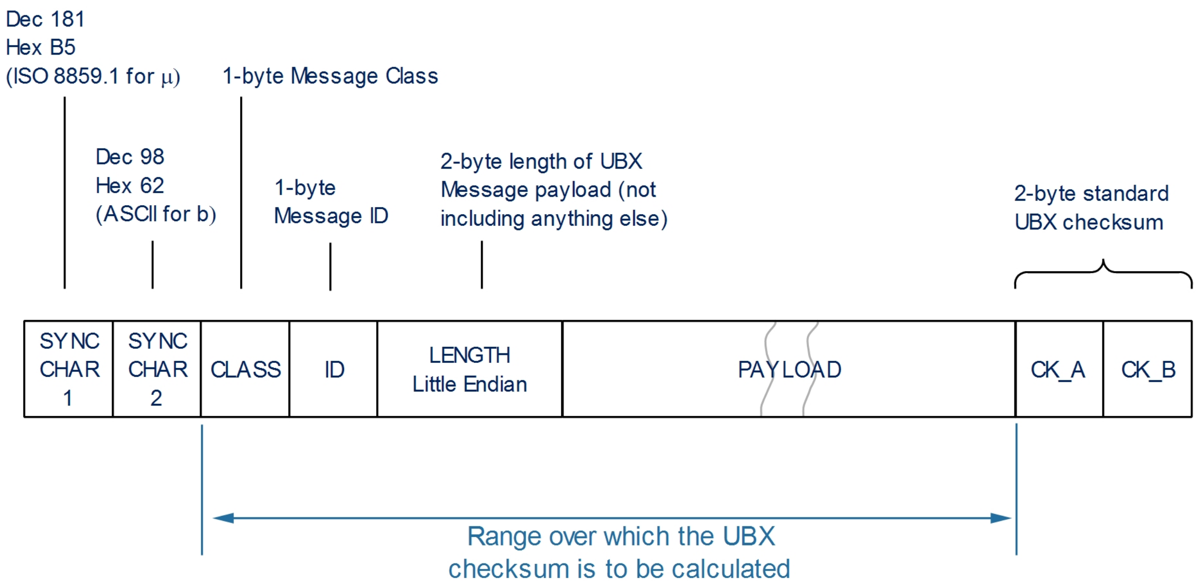
Short	Type	Size (Bytes)	Comment	Min/Max	Resolution
U1	Unsigned Char	1		0..255	1
RU1_3	Unsigned Char	1	binary floating point with 3 bit exponent, eeeb bbbb, (Value & 0x1F) << (Value >> 5)	0..(31*2^7) non-continuous	~ 2^(Value >> 5)
I1	Signed Char	1	2's complement	-128 .. 127	1
X1	Bitfield	1		n/a	n/a
U2	Unsigned Short	2		0 .. 65535	1
I2	Signed Short	2	2's complement	-32768 .. 32767	1
X2	Bitfield	2		n/a	n/a
U4	Unsigned Long	4		0 .. 4'294'967'295	1
I4	Signed Long	4	2's complement	-2'147'483'648 .. 2'147'483'647	1

Variable Type Definitions continued

Short	Type	Size (Bytes)	Comment	Min/Max	Resolution
X4	Bitfield	4		n/a	n/a
R4	IEEE 754 Single Precision	4		-1*2^+127 .. 2^+127	~ Value * 2^-24
R8	IEEE 754 Double Precision	8		-1*2^+1023 .. 2^+1023	~ Value * 2^-53
CH	ASCII / ISO 8859.1 Encoding	1			

25.4 UBX Checksum

The checksum is calculated over the Message, starting and including the CLASS field, up until, but excluding, the Checksum Field:



The checksum algorithm used is the 8-Bit Fletcher Algorithm, which is used in the TCP standard (RFC 1145). This algorithm works as follows:

- Buffer[N] contains the data over which the checksum is to be calculated.
- The two CK_ values are 8-Bit unsigned integers, only! If implementing with larger-sized integer values, make sure to mask both CK_A and CK_B with 0xFF after both operations in the loop.

```
CK_A = 0, CK_B = 0
For(I=0;I<N;I++)
{
    CK_A = CK_A + Buffer[I]
    CK_B = CK_B + CK_A
}
```

- After the loop, the two U1 values contain the checksum, transmitted after the Message, which conclude the Frame.

25.5 UBX Message Flow

There are certain features associated with the messages being sent back and forth:

25.5.1 Acknowledgement

When messages from the class CFG are sent to the receiver, the receiver will send an "acknowledge" (UBX-ACK-ACK) or a "not acknowledge" (UBX-ACK-NAK) message back to the sender, depending on whether or not the message was processed correctly.

Some messages from other classes (e.g. LOG) also use the same acknowledgement mechanism.

25.5.2 Polling Mechanism

All messages that are output by the receiver in a periodic manner (i.e. messages in classes MON, NAV and RXM) and Get/Set type messages, such as the configuration messages in the CFG class, can also be polled.

The UBX protocol is designed so that messages can be polled by sending the message required to the receiver but without a payload (or with just a single parameter that identifies the poll request). The receiver then responds with the same message with the payload populated.

25.6 UBX Class IDs

A class is a grouping of messages which are related to each other. The following table lists all the current message classes.

Name	Class	Description
NAV	0x01	Navigation Results Messages: Position, Speed, Time, Acceleration, Heading, DOP, SVs used
RXM	0x02	Receiver Manager Messages: Satellite Status, RTC Status
INF	0x04	Information Messages: Printf-Style Messages, with IDs such as Error, Warning, Notice
ACK	0x05	Ack/Nak Messages: Acknowledge or Reject messages to UBX-CFG input messages
CFG	0x06	Configuration Input Messages: Set Dynamic Model, Set DOP Mask, Set Baud Rate, etc.
UPD	0x09	Firmware Update Messages: Memory/Flash erase/write, Reboot, Flash identification, etc.
MON	0x0A	Monitoring Messages: Communication Status, CPU Load, Stack Usage, Task Status
AID	0x0B	AssistNow Aiding Messages: Ephemeris, Almanac, other A-GPS data input
TIM	0x0D	Timing Messages: Time Pulse Output, Time Mark Results
MGA	0x13	Multiple GNSS Assistance Messages: Assistance data for various GNSS
LOG	0x21	Logging Messages: Log creation, deletion, info and retrieval
SEC	0x27	Security Feature Messages

All remaining class IDs are reserved.

25.7 UBX Messages Overview

Page	Mnemonic	Cls/ID	Length	Type	Description
	UBX Class ACK			**Ack/Nak Messages**	
107	**ACK-ACK**	0x05 0x01	2	Output	Message Acknowledged
107	**ACK-NAK**	0x05 0x00	2	Output	Message Not-Acknowledged
	UBX Class AID			**AssistNow Aiding Messages**	
108	**AID-ALM**	0x0B 0x30	0	Poll Request	Poll GPS Aiding Almanac Data
108	**AID-ALM**	0x0B 0x30	1	Poll Request	Poll GPS Aiding Almanac Data for a SV
109	**AID-ALM**	0x0B 0x30	(8) or (40)	Input/Output	GPS Aiding Almanac Input/Output...
109	**AID-AOP**	0x0B 0x33	0	Poll Request	Poll AssistNow Autonomous data, all...
110	**AID-AOP**	0x0B 0x33	1	Poll Request	Poll AssistNow Autonomous data, one...
110	**AID-AOP**	0x0B 0x33	68	Input/Output	AssistNow Autonomous data
111	**AID-EPH**	0x0B 0x31	0	Poll Request	Poll GPS Aiding Ephemeris Data
111	**AID-EPH**	0x0B 0x31	1	Poll Request	Poll GPS Aiding Ephemeris Data for a SV
112	**AID-EPH**	0x0B 0x31	(8) or (104)	Input/Output	GPS Aiding Ephemeris Input/Output...
113	**AID-HUI**	0x0B 0x02	0	Poll Request	Poll GPS Health, UTC, ionosphere...
113	**AID-HUI**	0x0B 0x02	72	Input/Output	GPS Health, UTC and ionosphere...
115	**AID-INI**	0x0B 0x01	0	Poll Request	Poll GPS Initial Aiding Data
115	**AID-INI**	0x0B 0x01	48	Input/Output	Aiding position, time, frequency, clock...
	UBX Class CFG			**Configuration Input Messages**	
118	**CFG-ANT**	0x06 0x13	4	Get/Set	Antenna Control Settings
119	**CFG-BATCH**	0x06 0x93	8	Get/Set	Get/Set data batching configuration
120	**CFG-CFG**	0x06 0x09	(12) or (13)	Command	Clear, Save and Load configurations
122	**CFG-DAT**	0x06 0x06	44	Set	Set User-defined Datum.
123	**CFG-DAT**	0x06 0x06	52	Get	The currently defined Datum
124	**CFG-DYNSEED**	0x06 0x85	12	Set	Programming the dynamic seed for the...
124	**CFG-FIXSEED**	0x06 0x84	12 + 2*length	Set	Programming the fixed seed for host...
125	**CFG-GEOFENCE**	0x06 0x69	8 + 12*numF...	Get/Set	Geofencing configuration
126	**CFG-GNSS**	0x06 0x3E	4 + 8*numCo...	Get/Set	GNSS system configuration
128	**CFG-INF**	0x06 0x02	1	Poll Request	Poll configuration for one protocol
129	**CFG-INF**	0x06 0x02	0 + 10*N	Get/Set	Information message configuration
130	**CFG-ITFM**	0x06 0x39	8	Get/Set	Jamming/Interference Monitor...
131	**CFG-LOGFILTER**	0x06 0x47	12	Get/Set	Data Logger Configuration
133	**CFG-MSG**	0x06 0x01	2	Poll Request	Poll a message configuration
133	**CFG-MSG**	0x06 0x01	8	Get/Set	Set Message Rate(s)
134	**CFG-MSG**	0x06 0x01	3	Get/Set	Set Message Rate
134	**CFG-NAV5**	0x06 0x24	36	Get/Set	Navigation Engine Settings
136	**CFG-NAVX5**	0x06 0x23	40	Get/Set	Navigation Engine Expert Settings
139	**CFG-NMEA**	0x06 0x17	4	Get/Set	NMEA protocol configuration...

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
140	CFG-NMEA	0x06 0x17	12	Get/Set	NMEA protocol configuration V0...
143	CFG-NMEA	0x06 0x17	20	Get/Set	Extended NMEA protocol configuration V1
146	CFG-ODO	0x06 0x1E	20	Get/Set	Odometer, Low-speed COG Engine...
147	CFG-PM2	0x06 0x3B	48	Get/Set	Extended Power Management...
150	CFG-PMS	0x06 0x86	8	Get/Set	Power Mode Setup
151	CFG-PRT	0x06 0x00	1	Poll Request	Polls the configuration for one I/O Port
151	CFG-PRT	0x06 0x00	20	Get/Set	Port Configuration for UART
154	CFG-PRT	0x06 0x00	20	Get/Set	Port Configuration for SPI Port
157	CFG-PRT	0x06 0x00	20	Get/Set	Port Configuration for DDC Port
159	CFG-PWR	0x06 0x57	8	Set	Put receiver in a defined power state.
160	CFG-RATE	0x06 0x08	6	Get/Set	Navigation/Measurement Rate Settings
161	CFG-RINV	0x06 0x34	1 + 1*N	Get/Set	Contents of Remote Inventory
162	CFG-RST	0x06 0x04	4	Command	Reset Receiver / Clear Backup Data...
163	CFG-RXM	0x06 0x11	2	Get/Set	RXM configuration
163	CFG-SBAS	0x06 0x16	8	Get/Set	SBAS Configuration
165	CFG-USB	0x06 0x1B	108	Get/Set	USB Configuration
UBX Class INF				**Information Messages**	
167	INF-DEBUG	0x04 0x04	0 + 1*N	Output	ASCII output with debug contents
167	INF-ERROR	0x04 0x00	0 + 1*N	Output	ASCII output with error contents
168	INF-NOTICE	0x04 0x02	0 + 1*N	Output	ASCII output with informational contents
168	INF-TEST	0x04 0x03	0 + 1*N	Output	ASCII output with test contents
168	INF-WARNING	0x04 0x01	0 + 1*N	Output	ASCII output with warning contents
UBX Class LOG				**Logging Messages**	
170	LOG-BATCH	0x21 0x11	100	Polled	Batched data
173	LOG-CREATE	0x21 0x07	8	Command	Create Log File
174	LOG-ERASE	0x21 0x03	0	Command	Erase Logged Data
174	LOG-FINDTIME	0x21 0x0E	12	Input	Find index of a log entry based on a...
175	LOG-FINDTIME	0x21 0x0E	8	Output	Response to FINDTIME request
175	LOG-INFO	0x21 0x08	0	Poll Request	Poll for log information
175	LOG-INFO	0x21 0x08	48	Output	Log information
177	LOG-RETRIEVEBA...	0x21 0x10	4	Command	Request batch data
178	LOG-RETRIEVEPO...	0x21 0x0f	32	Output	Odometer log entry
178	LOG-RETRIEVEPOS	0x21 0x0b	40	Output	Position fix log entry
179	LOG-RETRIEVEST...	0x21 0x0d	16 + 1*byteCo...	Output	Byte string log entry
180	LOG-RETRIEVE	0x21 0x09	12	Command	Request log data
181	LOG-STRING	0x21 0x04	0 + 1*N	Command	Store arbitrary string in on-board flash
UBX Class MGA				**Multiple GNSS Assistance Messages**	
182	MGA-ACK-DATA0	0x13 0x60	8	Output	Multiple GNSS Acknowledge message

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
183	MGA-ANO	0x13 0x20	76	Input	Multiple GNSS AssistNow Offline...
183	MGA-BDS-EPH	0x13 0x03	88	Input	BDS Ephemeris Assistance
185	MGA-BDS-ALM	0x13 0x03	40	Input	BDS Almanac Assistance
186	MGA-BDS-HEALTH	0x13 0x03	68	Input	BDS Health Assistance
186	MGA-BDS-UTC	0x13 0x03	20	Input	BDS UTC Assistance
187	MGA-BDS-IONO	0x13 0x03	16	Input	BDS Ionospheric Assistance
187	MGA-DBD	0x13 0x80	0	Poll Request	Poll the Navigation Database
188	MGA-DBD	0x13 0x80	12 + 1*N	Input/Output	Navigation Database Dump Entry
188	MGA-FLASH-DATA	0x13 0x21	6 + 1*size	Input	Transfer MGA-ANO data block to flash
189	MGA-FLASH-STOP	0x13 0x21	2	Input	Finish flashing MGA-ANO data
189	MGA-FLASH-ACK	0x13 0x21	6	Output	Acknowledge last FLASH-DATA or -STOP
190	MGA-GAL-EPH	0x13 0x02	76	Input	Galileo Ephemeris Assistance
192	MGA-GAL-ALM	0x13 0x02	32	Input	Galileo Almanac Assistance
193	MGA-GAL-TIMEO...	0x13 0x02	12	Input	Galileo GPS time offset assistance
193	MGA-GAL-UTC	0x13 0x02	20	Input	Galileo UTC Assistance
194	MGA-GLO-EPH	0x13 0x06	48	Input	GLONASS Ephemeris Assistance
195	MGA-GLO-ALM	0x13 0x06	36	Input	GLONASS Almanac Assistance
196	MGA-GLO-TIMEO...	0x13 0x06	20	Input	GLONASS Auxiliary Time Offset...
197	MGA-GPS-EPH	0x13 0x00	68	Input	GPS Ephemeris Assistance
198	MGA-GPS-ALM	0x13 0x00	36	Input	GPS Almanac Assistance
199	MGA-GPS-HEALTH	0x13 0x00	40	Input	GPS Health Assistance
199	MGA-GPS-UTC	0x13 0x00	20	Input	GPS UTC Assistance
200	MGA-GPS-IONO	0x13 0x00	16	Input	GPS Ionosphere Assistance
201	MGA-INI-POS_XYZ	0x13 0x40	20	Input	Initial Position Assistance
202	MGA-INI-POS_LLH	0x13 0x40	20	Input	Initial Position Assistance
202	MGA-INI-TIME_UTC	0x13 0x40	24	Input	Initial Time Assistance
203	MGA-INI-TIME_GN...	0x13 0x40	24	Input	Initial Time Assistance
205	MGA-INI-CLKD	0x13 0x40	12	Input	Initial Clock Drift Assistance
205	MGA-INI-FREQ	0x13 0x40	12	Input	Initial Frequency Assistance
206	MGA-INI-EOP	0x13 0x40	72	Input	Earth Orientation Parameters Assistance
207	MGA-QZSS-EPH	0x13 0x05	68	Input	QZSS Ephemeris Assistance
208	MGA-QZSS-ALM	0x13 0x05	36	Input	QZSS Almanac Assistance
209	MGA-QZSS-HEAL...	0x13 0x05	12	Input	QZSS Health Assistance
UBX Class MON				**Monitoring Messages**	
210	MON-BATCH	0x0A 0x32	12	Polled	Data batching buffer status
210	MON-GNSS	0x0A 0x28	8	Polled	Information message major GNSS...
212	MON-HW2	0x0A 0x0B	28	Periodic/Polled	Extended Hardware Status
213	MON-HW	0x0A 0x09	60	Periodic/Polled	Hardware Status

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
214	MON-IO	0x0A 0x02	0 + 20*N	Periodic/Polled	I/O Subsystem Status
215	MON-MSGPP	0x0A 0x06	120	Periodic/Polled	Message Parse and Process Status
216	MON-PATCH	0x0A 0x27	0	Poll Request	Poll Request for installed patches
216	MON-PATCH	0x0A 0x27	4 + 16*nEntries	Polled	Output information about installed...
217	MON-RXBUF	0x0A 0x07	24	Periodic/Polled	Receiver Buffer Status
217	MON-RXR	0x0A 0x21	1	Output	Receiver Status Information
218	MON-TXBUF	0x0A 0x08	28	Periodic/Polled	Transmitter Buffer Status
219	MON-VER	0x0A 0x04	0	Poll Request	Poll Receiver/Software Version
219	MON-VER	0x0A 0x04	40 + 30*N	Polled	Receiver/Software Version
UBX Class NAV				**Navigation Results Messages**	
220	NAV-AOPSTATUS	0x01 0x60	16	Periodic/Polled	AssistNow Autonomous Status
221	NAV-CLOCK	0x01 0x22	20	Periodic/Polled	Clock Solution
221	NAV-DGPS	0x01 0x31	16 + 12*numCh	Periodic/Polled	DGPS Data Used for NAV
222	NAV-DOP	0x01 0x04	18	Periodic/Polled	Dilution of precision
223	NAV-EOE	0x01 0x61	4	Periodic	End Of Epoch
223	NAV-GEOFENCE	0x01 0x39	8 + 2*numFe...	Periodic/Polled	Geofencing status
224	NAV-HPPOSECEF	0x01 0x13	28	Periodic/Polled	High Precision Position Solution in ECEF
225	NAV-HPPOSLLH	0x01 0x14	36	Periodic/Polled	High Precision Geodetic Position Solution
226	NAV-ODO	0x01 0x09	20	Periodic/Polled	Odometer Solution
226	NAV-ORB	0x01 0x34	8 + 6*numSv	Periodic/Polled	GNSS Orbit Database Info
229	NAV-POSECEF	0x01 0x01	20	Periodic/Polled	Position Solution in ECEF
229	NAV-POSLLH	0x01 0x02	28	Periodic/Polled	Geodetic Position Solution
230	NAV-PVT	0x01 0x07	92	Periodic/Polled	Navigation Position Velocity Time...
232	NAV-RESETODO	0x01 0x10	0	Command	Reset odometer
233	NAV-SAT	0x01 0x35	8 + 12*numSvs	Periodic/Polled	Satellite Information
235	NAV-SBAS	0x01 0x32	12 + 12*cnt	Periodic/Polled	SBAS Status Data
236	NAV-SOL	0x01 0x06	52	Periodic/Polled	Navigation Solution Information
237	NAV-STATUS	0x01 0x03	16	Periodic/Polled	Receiver Navigation Status
240	NAV-SVINFO	0x01 0x30	8 + 12*numCh	Periodic/Polled	Space Vehicle Information
242	NAV-TIMEBDS	0x01 0x24	20	Periodic/Polled	BDS Time Solution
243	NAV-TIMEGAL	0x01 0x25	20	Periodic/Polled	Galileo Time Solution
244	NAV-TIMEGLO	0x01 0x23	20	Periodic/Polled	GLO Time Solution
245	NAV-TIMEGPS	0x01 0x20	16	Periodic/Polled	GPS Time Solution
246	NAV-TIMELS	0x01 0x26	24	Periodic/Polled	Leap second event information
248	NAV-TIMEUTC	0x01 0x21	20	Periodic/Polled	UTC Time Solution
249	NAV-VELECEF	0x01 0x11	20	Periodic/Polled	Velocity Solution in ECEF
250	NAV-VELNED	0x01 0x12	36	Periodic/Polled	Velocity Solution in NED
UBX Class RXM				**Receiver Manager Messages**	

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
251	**RXM-IMES**	0x02 0x61	4 + 44*numTx	Periodic/Polled	Indoor Messaging System Information
254	**RXM-MEASX**	0x02 0x14	44 + 24*num...	Periodic	Satellite Measurements for RRLP
255	**RXM-PMREQ**	0x02 0x41	8	Command	Requests a Power Management task
256	**RXM-PMREQ**	0x02 0x41	16	Command	Requests a Power Management task
257	**RXM-RLM**	0x02 0x59	16	Output	Galileo SAR Short-RLM report
258	**RXM-RLM**	0x02 0x59	28	Output	Galileo SAR Long-RLM report
258	**RXM-SFRBX**	0x02 0x13	8 + 4*numW...	Output	Broadcast Navigation Data Subframe
259	**RXM-SVSI**	0x02 0x20	8 + 6*numSV	Periodic/Polled	SV Status Info
UBX Class SEC				**Security Feature Messages**	
261	**SEC-SIGN**	0x27 0x01	40	Output	Signature of a previous message
261	**SEC-UNIQID**	0x27 0x03	9	Output	Unique Chip ID
UBX Class TIM				**Timing Messages**	
262	**TIM-TM2**	0x0D 0x03	28	Periodic/Polled	Time mark data
263	**TIM-VRFY**	0x0D 0x06	20	Periodic/Polled	Sourced Time Verification
UBX Class UPD				**Firmware Update Messages**	
265	**UPD-SOS**	0x09 0x14	0	Poll Request	Poll Backup File Restore Status
265	**UPD-SOS**	0x09 0x14	4	Command	Create Backup File in Flash
266	**UPD-SOS**	0x09 0x14	4	Command	Clear Backup in Flash
266	**UPD-SOS**	0x09 0x14	8	Output	Backup File Creation Acknowledge
267	**UPD-SOS**	0x09 0x14	8	Output	System Restored from Backup

25.8 UBX-ACK (0x05)

Ack/Nak Messages: i.e. Acknowledge or Reject messages to UBX-CFG input messages.
Messages in the UBX-ACK class output the processing results to UBX-CFG and some other messages.

25.8.1 UBX-ACK-ACK (0x05 0x01)

25.8.1.1 Message Acknowledged

Message	**UBX-ACK-ACK**				
Description	**Message Acknowledged**				
Type	Output				
Comment	Output upon processing of an input message. ACK Message is sent as soon as possible but at least within one second.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x05	0x01	2	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`clsID`	-	Class ID of the Acknowledged Message
1	U1	-	`msgID`	-	Message ID of the Acknowledged Message

25.8.2 UBX-ACK-NAK (0x05 0x00)

25.8.2.1 Message Not-Acknowledged

Message	**UBX-ACK-NAK**				
Description	**Message Not-Acknowledged**				
Type	Output				
Comment	Output upon processing of an input message. NAK Message is sent as soon as possible but at least within one second.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x05	0x00	2	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`clsID`	-	Class ID of the Not-Acknowledged Message
1	U1	-	`msgID`	-	Message ID of the Not-Acknowledged Message

25.9 UBX-AID (0x0B)

AssistNow Aiding Messages: i.e. Ephemeris, Almanac, other A-GPS data input.
Messages in the AID class are used to send GPS aiding data to the receiver.

25.9.1 UBX-AID-ALM (0x0B 0x30)

25.9.1.1 Poll GPS Aiding Almanac Data

Message	**UBX-AID-ALM**					
Description	**Poll GPS Aiding Almanac Data**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll GPS Aiding Data (Almanac) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type AID-ALM as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x30	0	see below	CK_A CK_B
No payload						

25.9.1.2 Poll GPS Aiding Almanac Data for a SV

Message	**UBX-AID-ALM**					
Description	**Poll GPS Aiding Almanac Data for a SV**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll GPS Aiding Data (Almanac) for an SV by sending this message to the receiver. The receiver will return one message of type AID-ALM as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x30	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`svid`	-	SV ID for which the receiver shall return its Almanac Data (Valid Range: 1 .. 32 or 51, 56, 63).

25.9.1.3 GPS Aiding Almanac Input/Output Message

Message	**UBX-AID-ALM**						
Description	**GPS Aiding Almanac Input/Output Message**						
Type	Input/Output						
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** • If the WEEK Value is 0, DWRD0 to DWRD7 are not sent as the Almanac is not available for the given SV. This may happen even if NAV-SVINFO and RXM-SVSI are indicating almanac availability as the internal data may not represent the content of an original broadcast almanac (or only parts thereof). • DWORD0 to DWORD7 contain the 8 words following the Hand-Over Word (HOW) from the GPS navigation message, either pages 1 to 24 of sub-frame 5 or pages 2 to 10 of subframe 4. See IS-GPS-200 for a full description of the contents of the Almanac pages. • In DWORD0 to DWORD7, the parity bits have been removed, and the 24 bits of data are located in Bits 0 to 23. Bits 24 to 31 shall be ignored. • Example: Parameter e (Eccentricity) from Almanac Subframe 4/5, Word 3, Bits 69-84 within the subframe can be found in DWRD0, Bits 15-0 whereas Bit 0 is the LSB.						
	Header	Class	ID	Length (Bytes)	Payload	Checksum	
Message Structure	0xB5 0x62	0x0B	0x30	(8) or (40)	see below	CK_A CK_B	

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	svid	-	SV ID for which this Almanac Data is (Valid Range: 1 .. 32 or 51, 56, 63).
4	U4	-	week	-	Issue Date of Almanac (GPS week number)
Start of optional block					
8	U4[8]	-	dwrd	-	Almanac Words
End of optional block					

25.9.2 UBX-AID-AOP (0x0B 0x33)

25.9.2.1 Poll AssistNow Autonomous data, all satellites

Message	**UBX-AID-AOP**					
Description	**Poll AssistNow Autonomous data, all satellites**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll AssistNow Autonomous aiding data for all GPS satellites by sending this empty message. The receiver will return an AID-AOP message (see definition below) for each GPS satellite for which data is available.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x33	0	see below	CK_A CK_B
No payload						

25.9.2.2 Poll AssistNow Autonomous data, one GPS satellite

Message	**UBX-AID-AOP**				
Description	**Poll AssistNow Autonomous data, one GPS satellite**				
Type	Poll Request				
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll the AssistNow Autonomous data for the specified GPS satellite. The receiver will return a AID-AOP message (see definition below) if data is available for the requested satellite.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x33	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`svid`	-	GPS SV ID for which the data is requested (valid range: 1..32).

25.9.2.3 AssistNow Autonomous data

Message	**UBX-AID-AOP**				
Description	**AssistNow Autonomous data**				
Type	Input/Output				
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** If enabled, this message is output at irregular intervals. It is output whenever AssistNow Autonomous has produced new data for a satellite. Depending on the availability of the optional data the receiver will output either version of the message. If this message is polled using one of the two poll requests described above the receiver will send this message if AssistNow Autonomous data is available or the corresponding poll request message if no AssistNow Autonomous data is available for each satellite (i.e. svid 1..32). At the user's choice the optional data may be chopped from the payload of a previously polled message when sending the message back to the receiver. Sending a valid AID-AOP message to the receiver will automatically enable the AssistNow Autonomous feature on the receiver. See the section AssistNow Autonomous in the receiver description for details on this feature.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x33	68	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`gnssId`	-	GNSS identifier (see Satellite Numbering)
1	U1	-	`svId`	-	Satellite identifier (see Satellite Numbering)
2	U1[2]	-	`reserved1`	-	Reserved
4	U1[64]	-	`data`	-	assistance data

25.9.3 UBX-AID-EPH (0x0B 0x31)

25.9.3.1 Poll GPS Aiding Ephemeris Data

Message	**UBX-AID-EPH**					
Description	**Poll GPS Aiding Ephemeris Data**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll GPS Aiding Data (Ephemeris) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type AID-EPH as defined below.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x31	0	see below	CK_A CK_B
No payload						

25.9.3.2 Poll GPS Aiding Ephemeris Data for a SV

Message	**UBX-AID-EPH**					
Description	**Poll GPS Aiding Ephemeris Data for a SV**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** Poll GPS Constellation Data (Ephemeris) for an SV by sending this message to the receiver. The receiver will return one message of type AID-EPH as defined below.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x31	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	svid	-	SV ID for which the receiver shall return its Ephemeris Data (Valid Range: 1 .. 32).

25.9.3.3 GPS Aiding Ephemeris Input/Output Message

Message	**UBX-AID-EPH**					
Description	**GPS Aiding Ephemeris Input/Output Message**					
Type	Input/Output					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** • SF1D0 to SF3D7 is only sent if ephemeris is available for this SV. If not, the payload may be reduced to 8 Bytes, or all bytes are set to zero, indicating that this SV Number does not have valid ephemeris for the moment. This may happen even if NAV-SVINFO and RXM-SVSI are indicating ephemeris availability as the internal data may not represent the content of an original broadcast ephemeris (or only parts thereof). • SF1D0 to SF3D7 contain the 24 words following the Hand-Over Word (HOW) from the GPS navigation message, subframes 1 to 3. The Truncated TOW Count is not valid and cannot be used. See IS-GPS-200 for a full description of the contents of the Subframes. • In SF1D0 to SF3D7, the parity bits have been removed, and the 24 bits of data are located in Bits 0 to 23. Bits 24 to 31 shall be ignored. • When polled, the data contained in this message does not represent the full original ephemeris broadcast. Some fields that are irrelevant to u-blox receivers may be missing. The week number in Subframe 1 has already been modified to match the Time Of Ephemeris (TOE).					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x31	(8) or (104)	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`svid`	-	SV ID for which this ephemeris data is (Valid Range: 1 .. 32).
4	U4	-	`how`	-	Hand-Over Word of first Subframe. This is required if data is sent to the receiver. 0 indicates that no Ephemeris Data is following.
Start of optional block					
8	U4[8]	-	`sf1d`	-	Subframe 1 Words 3..10 (SF1D0..SF1D7)
40	U4[8]	-	`sf2d`	-	Subframe 2 Words 3..10 (SF2D0..SF2D7)
72	U4[8]	-	`sf3d`	-	Subframe 3 Words 3..10 (SF3D0..SF3D7)
End of optional block					

25.9.4 UBX-AID-HUI (0x0B 0x02)

25.9.4.1 Poll GPS Health, UTC, ionosphere parameters

Message	**UBX-AID-HUI**					
Description	**Poll GPS Health, UTC, ionosphere parameters**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** -					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x02	0	see below	CK_A CK_B
No payload						

25.9.4.2 GPS Health, UTC and ionosphere parameters

Message	**UBX-AID-HUI**					
Description	**GPS Health, UTC and ionosphere parameters**					
Type	Input/Output					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** This message contains a health bit mask, UTC time and Klobuchar parameters. For more information on these parameters, see the ICD-GPS-200 documentation.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0B	0x02	72	see below	CK_A CK_B

Payload Contents:

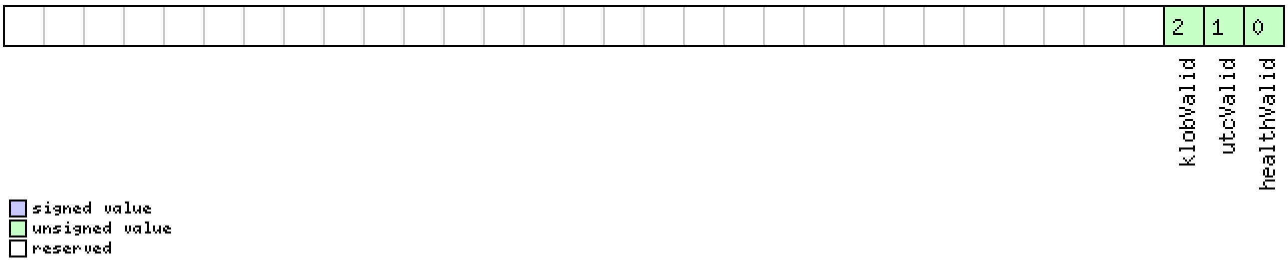
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X4	-	health	-	Bitmask, every bit represenst a GPS SV (1-32). If the bit is set the SV is healthy.
4	R8	-	utcA0	-	UTC - parameter A0
12	R8	-	utcA1	-	UTC - parameter A1
20	I4	-	utcTOW	-	UTC - reference time of week
24	I2	-	utcWNT	-	UTC - reference week number
26	I2	-	utcLS	-	UTC - time difference due to leap seconds before event
28	I2	-	utcWNF	-	UTC - week number when next leap second event occurs
30	I2	-	utcDN	-	UTC - day of week when next leap second event occurs
32	I2	-	utcLSF	-	UTC - time difference due to leap seconds after event
34	I2	-	utcSpare	-	UTC - Spare to ensure structure is a multiple of 4 bytes
36	R4	-	klobA0	s	Klobuchar - alpha 0
40	R4	-	klobA1	s/semi circle	Klobuchar - alpha 1

UBX-AID-HUI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
44	R4	-	klobA2	s/semi circle^2	Klobuchar - alpha 2
48	R4	-	klobA3	s/semi circle^3	Klobuchar - alpha 3
52	R4	-	klobB0	s	Klobuchar - beta 0
56	R4	-	klobB1	s/semi circle	Klobuchar - beta 1
60	R4	-	klobB2	s/semi circle^2	Klobuchar - beta 2
64	R4	-	klobB3	s/semi circle^3	Klobuchar - beta 3
68	X4	-	flags	-	flags (see graphic below)

Bitfield flags

This graphic explains the bits of flags



Name	Description
healthValid	Healthmask field in this message is valid
utcValid	UTC parameter fields in this message are valid
klobValid	Klobuchar parameter fields in this message are valid

25.9.5 UBX-AID-INI (0x0B 0x01)

25.9.5.1 Poll GPS Initial Aiding Data

Message	**UBX-AID-INI**					
Description	**Poll GPS Initial Aiding Data**					
Type	Poll Request					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x01	0	see below	CK_A CK_B
No payload						

25.9.5.2 Aiding position, time, frequency, clock drift

Message	**UBX-AID-INI**					
Description	**Aiding position, time, frequency, clock drift**					
Type	Input/Output					
Comment	**All UBX-AID messages are deprecated; use UBX-MGA messages instead** This message contains position, time and clock drift information. The position can be input in either the ECEF X/Y/Z coordinate system or as lat/lon/height. The time can either be input as inexact value via the standard communication interface, suffering from latency depending on the baud rate, or using hardware time synchronization where an accurate time pulse is input on the external interrupts. It is also possible to supply hardware frequency aiding by connecting a continuous signal to an external interrupt.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x01	48	see below	CK_A CK_B

Payload Contents:

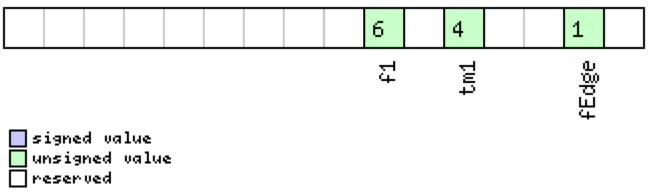
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	I4	-	ecefXOrLat	cm_or_deg*1e-7	WGS84 ECEF X coordinate or latitude, depending on flags below
4	I4	-	ecefYOrLon	cm_or_deg*1e-7	WGS84 ECEF Y coordinate or longitude, depending on flags below
8	I4	-	ecefZOrAlt	cm	WGS84 ECEF Z coordinate or altitude, depending on flags below
12	U4	-	posAcc	cm	Position accuracy (stddev)
16	X2	-	tmCfg	-	Time mark configuration (see graphic below)
18	U2	-	wnoOrDate	week_or_yearMonth	Actual week number or yearSince2000/Month (YYMM), depending on flags below

UBX-AID-INI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
20	U4	-	towOrTime	ms_or_dayHourMinuteSec	Actual time of week or DayOfMonth/Hour/Minute/Second (DDHHMMSS), depending on flags below
24	I4	-	towNs	ns	Fractional part of time of week
28	U4	-	tAccMs	ms	Milliseconds part of time accuracy
32	U4	-	tAccNs	ns	Nanoseconds part of time accuracy
36	I4	-	clkDOrFreq	ns/s_or_Hz*1e-2	Clock drift or frequency, depending on flags below
40	U4	-	clkDAccOrFreqAcc	ns/s_or_ppb	Accuracy of clock drift or frequency, depending on flags below
44	X4	-	flags	-	Bitmask with the following flags (see graphic below)

Bitfield tmCfg

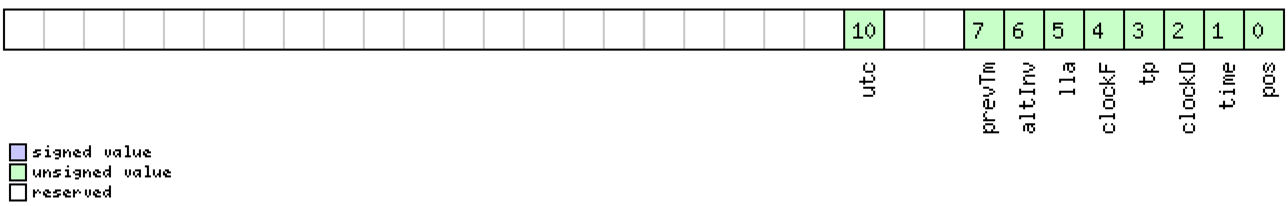
This graphic explains the bits of tmCfg



Name	Description
fEdge	use falling edge (default rising)
tm1	time mark on extint 1 (default extint 0)
f1	frequency on extint 1 (default extint 0)

Bitfield flags

This graphic explains the bits of flags



Name	Description
pos	Position is valid
time	Time is valid
clockD	Clock drift data contains valid clock drift, must not be set together with clockF
tp	Use time pulse
clockF	Clock drift data contains valid frequency, must not be set together with clockD
lla	Position is given in lat/long/alt (default is ECEF)
altInv	Altitude is not valid, if lla was set
prevTm	Use time mark received before AID-INI message (default uses mark received after message)
utc	Time is given as UTC date/time (default is GPS wno/tow)

25.10 UBX-CFG (0x06)

Configuration Input Messages: i.e. Set Dynamic Model, Set DOP Mask, Set Baud Rate, etc.. Messages in the CFG class are used to configure the receiver and read out current configuration values. Any messages in the CFG class sent to the receiver are either acknowledged (with message UBX-ACK-ACK) if processed successfully or rejected (with message UBX-ACK-NAK) if processing unsuccessfully.

25.10.1 UBX-CFG-ANT (0x06 0x13)

25.10.1.1 Antenna Control Settings

Message	**UBX-CFG-ANT**
Description	**Antenna Control Settings**
Type	Get/Set
Comment	This message allows the user to configure the antenna supervisor. The antenna supervisor can be used to detect the status of an active antenna and control it. It can be used to turn off the supply to the antenna in the event of a short (for example) or to manage power consumption in Power Save Mode. Refer to Antenna Supervisor Configuration and the relevant Hardware Integration Manual (HIM) for more information regarding the behavior of the antenna supervisor. Refer to UBX-MON-HW for a description of the fields in the message used to obtain the status of the antenna. Note that not all pins can be used for antenna supervisor operation, it is recommended that you use the default pins, consult the Integration Manual if you need to use other pins.

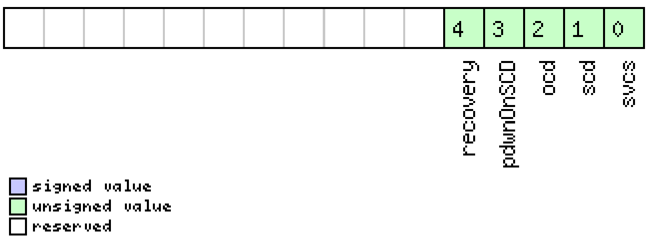
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x13	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X2	-	flags	-	Antenna Flag Mask (see graphic below)
2	X2	-	pins	-	Antenna Pin Configuration (see graphic below)

Bitfield flags

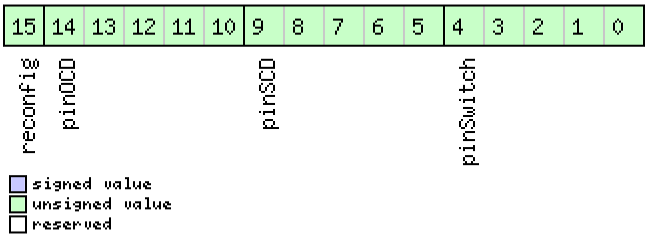
This graphic explains the bits of flags



Name	Description
svcs	Enable Antenna Supply Voltage Control Signal
scd	Enable Short Circuit Detection
ocd	Enable Open Circuit Detection
pdwnOnSCD	Power Down Antenna supply if Short Circuit is detected. (only in combination with Bit 1)
recovery	Enable automatic recovery from short state

Bitfield pins

This graphic explains the bits of pins



Name	Description
pinSwitch	PIO-Pin used for switching antenna supply
pinSCD	PIO-Pin used for detecting a short in the antenna supply
pinOCD	PIO-Pin used for detecting open/not connected antenna
reconfig	if set to one, and this command is sent to the receiver, the receiver will reconfigure the pins as specified.

25.10.2 UBX-CFG-BATCH (0x06 0x93)

25.10.2.1 Get/Set data batching configuration

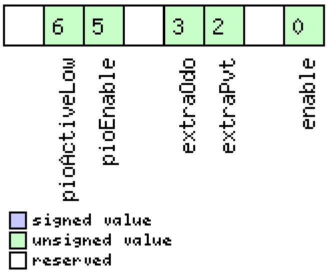
Message	**UBX-CFG-BATCH**					
Description	**Get/Set data batching configuration**					
Type	Get/Set					
Comment	Gets or sets the configuration for data batching. See Data Batching for more information.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x93	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x00 for this version)
1	X1	-	flags	-	Flags (see graphic below)
2	U2	-	bufSize	-	Size of buffer in number of epochs to store
4	U2	-	notifThrs	-	Buffer fill level that triggers PIO notification, in number of epochs stored
6	U1	-	pioId	-	PIO ID to use for buffer level notification
7	U1	-	reserved1	-	Reserved

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
`enable`	Enable data batching
`extraPvt`	Store extra PVT information The fields `iTOW`, `tAcc`, `numSV`, `hMSL`, `vAcc`, `velN`, `velE`, `velD`, `sAcc`, `headAcc` and `pDOP` in UBX-LOG-BATCH are only valid if this flag is set.
`extraOdo`	Store odometer data The fields `distance`, `totalDistance` and `distanceStd` in UBX-LOG-BATCH are only valid if this flag is set. Note: the odometer feature itself must also be enabled.
`pioEnable`	Enable PIO notification
`pioActiveLow`	PIO is active low

25.10.3 UBX-CFG-CFG (0x06 0x09)

25.10.3.1 Clear, Save and Load configurations

Message	**UBX-CFG-CFG**					
Description	**Clear, Save and Load configurations**					
Type	Command					
Comment	See Receiver Configuration for a detailed description on how Receiver Configuration should be used. The three masks are made up of individual bits, each bit indicating the sub-section of all configurations on which the corresponding action shall be carried out. The reserved bits in the masks must be set to '0'. For detailed information refer to the Organization of the Configuration Sections. Note that commands can be combined. The sequence of execution is Clear, Save, Load.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x09	(12) or (13)	see below	CK_A CK_B

Payload Contents:

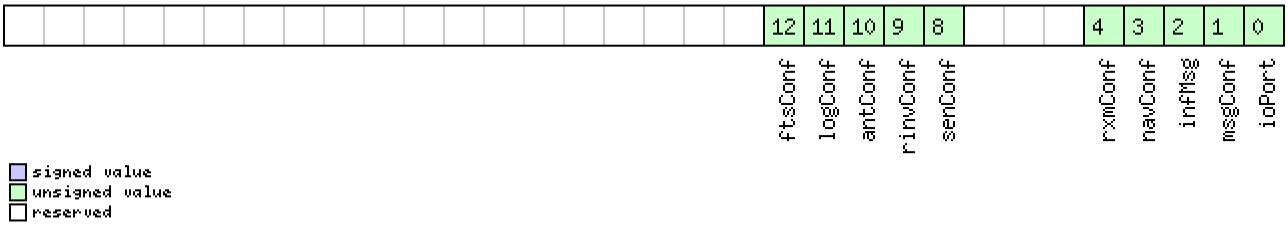
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X4	-	`clearMask`	-	Mask with configuration sub-sections to clear (i.e. load default configurations to permanent configurations in non-volatile memory) (see graphic below)
4	X4	-	`saveMask`	-	Mask with configuration sub-sections to save (i.e. save current configurations to non-volatile memory), see ID description of clearMask

UBX-CFG-CFG continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
8	X4	-	loadMask	-	Mask with configuration sub-sections to load (i.e. load permanent configurations from non-volatile memory to current configurations), see ID description of clearMask
Start of optional block					
12	X1	-	deviceMask	-	Mask which selects the memory devices for this command. (see graphic below)
End of optional block					

Bitfield clearMask

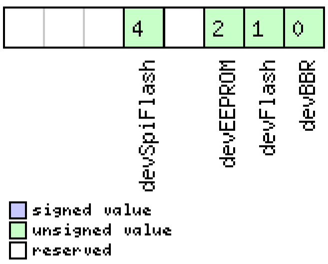
This graphic explains the bits of clearMask



Name	Description
ioPort	Communications port settings. Modifying this sub-section results in an IO system reset. Because of this undefined data may be output for a short period of time after receiving the message.
msgConf	Message configuration
infMsg	INF message configuration
navConf	Navigation configuration
rxmConf	Receiver Manager configuration
senConf	Sensor interface configuration
rinvConf	Remote inventory configuration
antConf	Antenna configuration
logConf	Logging configuration
ftsConf	FTS configuration. Only applicable to the FTS product variant.

Bitfield deviceMask

This graphic explains the bits of deviceMask



Name	Description
devBBR	Battery backed RAM
devFlash	Flash
devEEPROM	EEPROM
devSpiFlash	SPI Flash

25.10.4 UBX-CFG-DAT (0x06 0x06)

25.10.4.1 Set User-defined Datum.

Message	UBX-CFG-DAT					
Description	**Set User-defined Datum.**					
Type	Set					
Comment	For more information see the description of Geodetic Systems and Frames.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x06	44	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	R8	-	majA	m	Semi-major Axis (accepted range = 6,300, 000.0 to 6,500,000.0 meters).
8	R8	-	flat	-	1.0 / Flattening (accepted range is 0.0 to 500.0).
16	R4	-	dX	m	X Axis shift at the origin (accepted range is +/- 5000.0 meters).
20	R4	-	dY	m	Y Axis shift at the origin (accepted range is +/- 5000.0 meters).
24	R4	-	dZ	m	Z Axis shift at the origin (accepted range is +/- 5000.0 meters).
28	R4	-	rotX	s	Rotation about the X Axis (accepted range is +/- 20.0 milli-arc seconds).
32	R4	-	rotY	s	Rotation about the Y Axis (accepted range is +/- 20.0 milli-arc seconds).
36	R4	-	rotZ	s	Rotation about the Z Axis (accepted range is +/- 20.0 milli-arc seconds).
40	R4	-	scale	ppm	Scale change (accepted range is 0.0 to 50.0 parts per million).

25.10.4.2 The currently defined Datum

Message	**UBX-CFG-DAT**					
Description	**The currently defined Datum**					
Type	Get					
Comment	Returns the parameters of the currently defined datum. If no user-defined datum has been set, this will default to WGS84.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x06	52	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2	-	datumNum	-	Datum Number: 0 = WGS84, 0xFFFF = user-defined
2	CH[6]	-	datumName	-	ASCII String: WGS84 or USER
8	R8	-	majA	m	Semi-major Axis (accepted range = 6,300, 000.0 to 6,500,000.0 meters).
16	R8	-	flat	-	1.0 / Flattening (accepted range is 0.0 to 500.0).
24	R4	-	dX	m	X Axis shift at the origin (accepted range is +/- 5000.0 meters).
28	R4	-	dY	m	Y Axis shift at the origin (accepted range is +/- 5000.0 meters).
32	R4	-	dZ	m	Z Axis shift at the origin (accepted range is +/- 5000.0 meters).
36	R4	-	rotX	s	Rotation about the X Axis (accepted range is +/- 20.0 milli-arc seconds).
40	R4	-	rotY	s	Rotation about the Y Axis (accepted range is +/- 20.0 milli-arc seconds).
44	R4	-	rotZ	s	Rotation about the Z Axis (accepted range is +/- 20.0 milli-arc seconds).
48	R4	-	scale	ppm	Scale change (accepted range is 0.0 to 50.0 parts per million).

25.10.5 UBX-CFG-DYNSEED (0x06 0x85)

25.10.5.1 Programming the dynamic seed for the host interface signature

Message	**UBX-CFG-DYNSEED**					
Description	**Programming the dynamic seed for the host interface signature**					
Type	Set					
Comment	The message can be used to program the dynamic seed for the host interface signature. If successfully configured, the message will answer with ACK, otherwise with NAK. Before the first programming, it is assumed that the dynamic seed is all '0'.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x85	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x01 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	seedHi	-	high word of dynamic seed
8	U4	-	seedLo	-	low word of dynamic seed

25.10.6 UBX-CFG-FIXSEED (0x06 0x84)

25.10.6.1 Programming the fixed seed for host interface signature

Message	**UBX-CFG-FIXSEED**					
Description	**Programming the fixed seed for host interface signature**					
Type	Set					
Comment	The message can be used to program the fixed seed for the host interface signature. Moreover it will configure the set of messages that will be signed (min. 1, max. 10). If the class ID of the message is 0 the configuration is ignored for that message. If successfully configured, the message will answer with ACK, otherwise with NAK. See the configuring the fixed seed and register messages description for feature details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x84	12 + 2*length	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x02 for this version)
1	U1	-	length	-	Number of registered messages (min. 1, max. 10)
2	U1[2]	-	reserved1	-	Reserved
4	U4	-	seedHi	-	high word of fixed seed
8	U4	-	seedLo	-	low word of fixed seed
Start of repeated block (length times)					
12 + 2*N	U1	-	classId	-	Class ID on the message
13 + 2*N	U1	-	msgId	-	Message ID on the message

UBX-CFG-FIXSEED continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
End of repeated block					

25.10.7 UBX-CFG-GEOFENCE (0x06 0x69)

25.10.7.1 Geofencing configuration

Message	**UBX-CFG-GEOFENCE**					
Description	**Geofencing configuration**					
Type	Get/Set					
Comment	Gets or sets the geofencing configuration See the Geofencing description for feature details. If the receiver is sent a valid new configuration, it will respond with a UBX-ACK-ACK message and immediately change to the new configuration. Otherwise the receiver will reject the request, by issuing a UBX-ACK-NAK and continuing operation with the previous configuration. Note that the acknowledge message does not indicate whether the PIO configuration has been successfully applied (pin assigned), it only indicates the successful configuration of the feature. The configured PIO must be previously unoccupied for successful assignment.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x69	8 + 12*numFences	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (=0x00 for this version)
1	U1	-	numFences	-	Number of geofences contained in this message. Note that the receiver can only store a limited number of geofences (currently 4).
2	U1	-	confLvl	-	Required confidence level for state evaluation. This value times the position's standard deviation (sigma) defines the confidence band. 0 = no confidence required 1 = 68% 2 = 95% 3 = 99.7% 4 = 99.99%
3	U1[1]	-	reserved1	-	Reserved
4	U1	-	pioEnabled	-	1 = Enable PIO combined fence state output, 0 = disable
5	U1	-	pinPolarity	-	PIO pin polarity. 0 = Low means inside, 1 = Low means outside. Unknown state is always high.
6	U1	-	pin	-	PIO pin number

UBX-CFG-GEOFENCE continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
7	U1[1]	-	reserved2	-	Reserved
Start of repeated block (numFences times)					
8 + 12*N	I4	1e-7	lat	deg	Latitude of the geofence circle center
12 + 12*N	I4	1e-7	lon	deg	Longitude of the geofence circle center
16 + 12*N	U4	1e-2	radius	m	Radius of the geofence circle
End of repeated block					

25.10.8 UBX-CFG-GNSS (0x06 0x3E)

25.10.8.1 GNSS system configuration

Message	**UBX-CFG-GNSS**
Description	**GNSS system configuration**
Type	Get/Set
Comment	Gets or sets the GNSS system channel sharing configuration. If the receiver is sent a valid new configuration, it will respond with a UBX-ACK-ACK message and immediately change to the new configuration. Otherwise the receiver will reject the request, by issuing a UBX-ACK-NAK and continuing operation with the previous configuration. Configuration requirements: • It is necessary for at least one major GNSS to be enabled, after applying the new configuration to the current one. • It is also required that at least 4 tracking channels are available to each enabled major GNSS, i.e. maxTrkCh must have a minimum value of 4 for each enabled major GNSS. • The number of tracking channels in use must not exceed the number of tracking channels available in hardware, and the sum of all reserved tracking channels needs to be less than or equal to the number of tracking channels in use. Notes: • To avoid cross-correlation issues, it is recommended that GPS and QZSS are always both enabled or both disabled. • Polling this message returns the configuration of all supported GNSS, whether enabled or not; it may also include GNSS unsupported by the particular product, but in such cases the enable flag will always be unset. • See section GNSS Configuration for a discussion of the use of this message. • See section Satellite Numbering for a description of the GNSS IDs available. • Configuration specific to the GNSS system can be done via other messages (e.g. **UBX-CFG-SBAS**).

	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x3E	4 + 8*numConfigBlocks	see below	CK_A CK_B

Payload Contents:

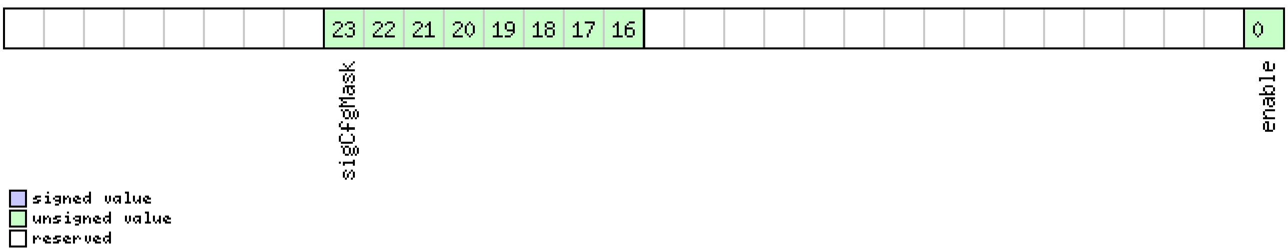
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	msgVer	-	Message version (=0 for this version)

UBX-CFG-GNSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
1	U1	-	numTrkChHw	-	Number of tracking channels available in hardware (read only)
2	U1	-	numTrkChUse	-	(Read only in protocol versions greater than 23) Number of tracking channels to use. Must be > 0, <= numTrkChHw. If 0xFF, then number of tracking channels to use will be set to numTrkChHw.
3	U1	-	numConfigBlocks	-	Number of configuration blocks following
Start of repeated block (numConfigBlocks times)					
4 + 8*N	U1	-	gnssId	-	System identifier (see Satellite Numbering)
5 + 8*N	U1	-	resTrkCh	-	(Read only in protocol versions greater than 23) Number of reserved (minimum) tracking channels for this system.
6 + 8*N	U1	-	maxTrkCh	-	(Read only in protocol versions greater than 23) Maximum number of tracking channels used for this system. Must be > 0, >= resTrkChn, <= numTrkChUse and <= maximum number of tracking channels supported for this system.
7 + 8*N	U1	-	reserved1	-	Reserved
8 + 8*N	X4	-	flags	-	bitfield of flags. At least one signal must be configured in every enabled system. (see graphic below)
End of repeated block					

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
enable	Enable this system
sigCfgMask	Signal configuration mask When gnssId is 0 (GPS) * 0x01 = GPS L1C/A * 0x10 = GPS L2C When gnssId is 1 (SBAS) * 0x01 = SBAS L1C/A When gnssId is 2 (Galileo) * 0x01 = Galileo E1 * 0x20 = Galileo E5b When gnssId is 3 (BeiDou) * 0x01 = BeiDou B1I * 0x10 = BeiDou B2I When gnssId is 4 (IMES) * 0x01 = IMES L1 When gnssId is 5 (QZSS) * 0x01 = QZSS L1C/A * 0x04 = QZSS L1S * 0x10 = QZSS L2C When gnssId is 6 (GLONASS) * 0x01 = GLONASS L1 * 0x10 = GLONASS L2

25.10.9 UBX-CFG-INF (0x06 0x02)

25.10.9.1 Poll configuration for one protocol

Message	**UBX-CFG-INF**					
Description	**Poll configuration for one protocol**					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x02	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	protocolID	-	Protocol Identifier, identifying the output protocol for this Poll Request. The following are valid Protocol Identifiers: 0: UBX Protocol 1: NMEA Protocol 2-255: Reserved

25.10.9.2 Information message configuration

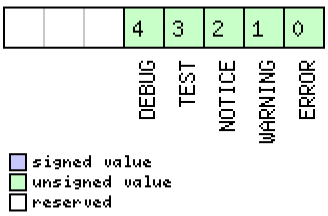
Message	**UBX-CFG-INF**					
Description	**Information message configuration**					
Type	Get/Set					
Comment	The value of infMsgMask[x] below are that each bit represents one of the INF class messages (Bit 0 for ERROR, Bit 1 for WARNING and so on.). For a complete list, see the Message Class INF. Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length. Output messages from the module contain only one configuration unit. Note that I/O Ports 1 and 2 correspond to serial ports 1 and 2. I/O port 0 is DDC. I/O port 3 is USB. I/O port 4 is SPI. I/O port 5 is reserved for future use.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x02	0 + 10*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*10	U1	-	`protocolID`	-	Protocol Identifier, identifying for which protocol the configuration is set/get. The following are valid Protocol Identifiers: 0: UBX Protocol 1: NMEA Protocol 2-255: Reserved
1 + 10*N	U1[3]	-	`reserved1`	-	Reserved
4 + 10*N	X1[6]	-	`infMsgMask`	-	A bit mask, saying which information messages are enabled on each I/O port (see graphic below)
End of repeated block					

Bitfield infMsgMask

This graphic explains the bits of `infMsgMask`



Name	Description
ERROR	enable ERROR
WARNING	enable WARNING
NOTICE	enable NOTICE
TEST	enable TEST
DEBUG	enable DEBUG

25.10.10 UBX-CFG-ITFM (0x06 0x39)

25.10.10.1 Jamming/Interference Monitor configuration

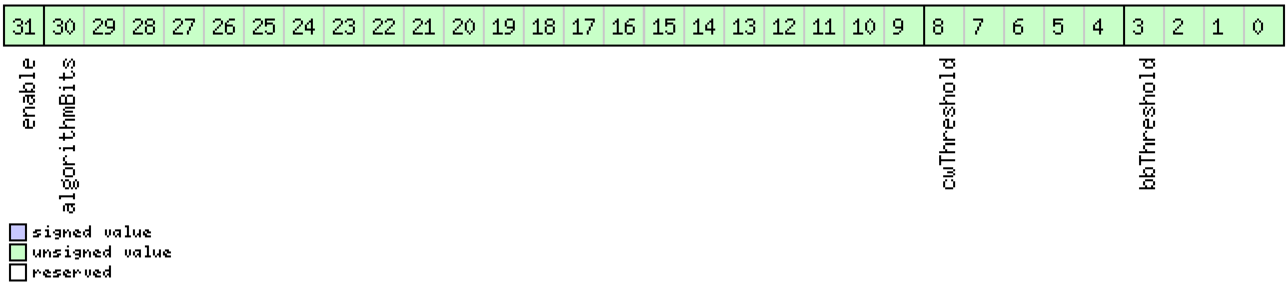
Message	**UBX-CFG-ITFM**				
Description	**Jamming/Interference Monitor configuration**				
Type	Get/Set				
Comment	Configuration of Jamming/Interference monitor.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x39	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X4	-	config	-	interference config word. (see graphic below)
4	X4	-	config2	-	extra settings for jamming/interference monitor (see graphic below)

Bitfield config

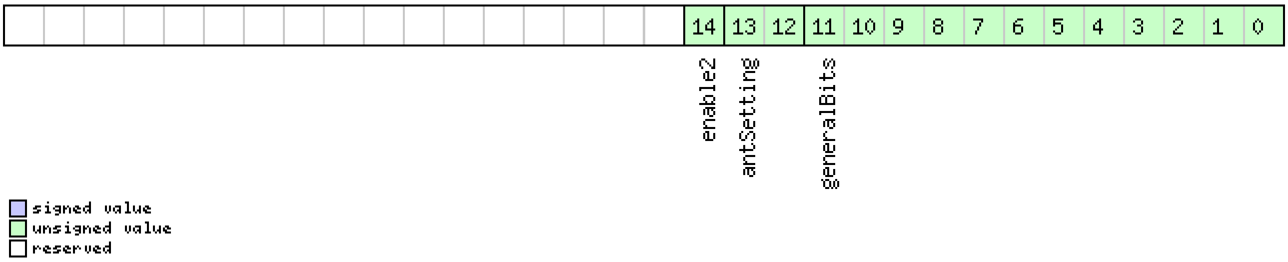
This graphic explains the bits of config



Name	Description
bbThreshold	Broadband jamming detection threshold (unit = dB)
cwThreshold	CW jamming detection threshold (unit = dB)
algorithmBits	reserved algorithm settings - should be set to 0x16B156 in hex for correct settings
enable	enable interference detection

Bitfield config2

This graphic explains the bits of `config2`



Name	Description
`generalBits`	general settings - should be set to 0x31E in hex for correct setting
`antSetting`	antennaSetting, 0=unknown, 1=passive, 2=active
`enable2`	Set to 1 to scan auxiliary bands (u-blox 8 / u-blox M8 only, otherwise ignored)

25.10.11 UBX-CFG-LOGFILTER (0x06 0x47)

25.10.11.1 Data Logger Configuration

Message	**UBX-CFG-LOGFILTER**					
Description	**Data Logger Configuration**					
Type	Get/Set					
Comment	This message can be used to configure the data logger, i.e. to enable/disable the log recording and to get/set the position entry filter settings. Position entries can be filtered based on time difference, position difference or current speed thresholds. Position and speed filtering also have a minimum time interval. A position is logged if any of the thresholds are exceeded. If a threshold is set to zero it is ignored. The maximum rate of position logging is 1Hz. The filter settings will be configured to the provided values only if the 'applyAllFilterSettings' flag is set. This allows the recording to be enabled/disabled independently of configuring the filter settings. It is supported to configure the data logger in the absence of a logging file. By doing so, once the logging file is created, the data logger configuration will take effect immediately and logging recording and filtering will activate according to the configuration.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x47	12	see below	CK_A CK_B

Payload Contents:

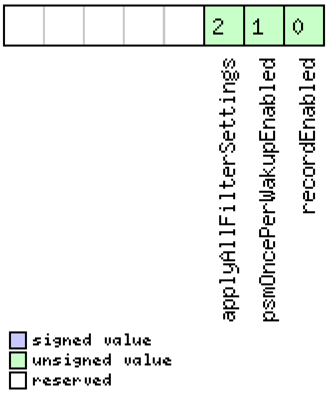
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	The version of this message. Set to 1
1	X1	-	`flags`	-	Flags (see graphic below)
2	U2	-	`minInterval`	s	Minimum time interval between logged positions (0 = not set). **This is only applied in combination with the speed and/or position thresholds.** If both minInterval and timeThreshold are set, minInterval must be less than or equal to timeThreshold.

UBX-CFG-LOGFILTER continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	U2	-	timeThreshold	s	If the time difference is greater than the threshold then the position is logged (0 = not set).
6	U2	-	speedThreshold	m/s	If the current speed is greater than the threshold then the position is logged (0 = not set). minInterval also applies
8	U4	-	positionThreshold	m	If the 3D position difference is greater than the threshold then the position is logged (0 = not set). minInterval also applies

Bitfield flags

This graphic explains the bits of flags



Name	Description
recordEnabled	1 = enable recording, 0 = disable recording
psmOncePerWakupEnabled	1 = enable recording only one single position per PSM on/off mode wake-up period, 0 = disable once per wake-up
applyAllFilterSettings	1 = apply all filter settings, 0 = only apply recordEnabled

25.10.12 UBX-CFG-MSG (0x06 0x01)

25.10.12.1 Poll a message configuration

Message	**UBX-CFG-MSG**					
Description	**Poll a message configuration**					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x01	2	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`msgClass`	-	Message Class
1	U1	-	`msgID`	-	Message Identifier

25.10.12.2 Set Message Rate(s)

Message	**UBX-CFG-MSG**					
Description	**Set Message Rate(s)**					
Type	Get/Set					
Comment	Set/Get message rate configuration (s) to/from the receiver. See also section How to change between protocols. • Send rate is relative to the event a message is registered on. For example, if the rate of a navigation message is set to 2, the message is sent every second navigation solution. For configuring NMEA messages, the section NMEA Messages Overview describes Class and Identifier numbers used.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x01	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`msgClass`	-	Message Class
1	U1	-	`msgID`	-	Message Identifier
2	U1[6]	-	`rate`	-	Send rate on I/O Port (6 Ports)

25.10.12.3 Set Message Rate

Message	**UBX-CFG-MSG**					
Description	**Set Message Rate**					
Type	Get/Set					
Comment	Set message rate configuration for the current port. See also section How to change between protocols.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x01	3	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`msgClass`	-	Message Class
1	U1	-	`msgID`	-	Message Identifier
2	U1	-	`rate`	-	Send rate on current Port

25.10.13 UBX-CFG-NAV5 (0x06 0x24)

25.10.13.1 Navigation Engine Settings

Message	**UBX-CFG-NAV5**					
Description	**Navigation Engine Settings**					
Type	Get/Set					
Comment	See the Navigation Configuration Settings Description for a detailed description of how these settings affect receiver operation.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x24	36	see below	CK_A CK_B

Payload Contents:

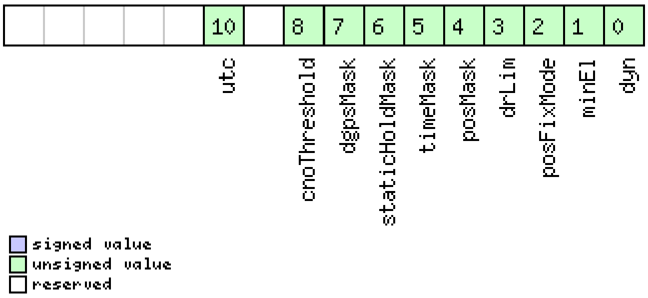
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X2	-	`mask`	-	Parameters Bitmask. Only the masked parameters will be applied. (see graphic below)
2	U1	-	`dynModel`	-	Dynamic platform model: 0: portable 2: stationary 3: pedestrian 4: automotive 5: sea 6: airborne with <1g acceleration 7: airborne with <2g acceleration 8: airborne with <4g acceleration 9: wrist worn watch 10: bike
3	U1	-	`fixMode`	-	Position Fixing Mode: 1: 2D only 2: 3D only 3: auto 2D/3D

UBX-CFG-NAV5 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	I4	0.01	fixedAlt	m	Fixed altitude (mean sea level) for 2D fix mode.
8	U4	0.0001	fixedAltVar	m^2	Fixed altitude variance for 2D mode.
12	I1	-	minElev	deg	Minimum Elevation for a GNSS satellite to be used in NAV
13	U1	-	drLimit	s	Reserved
14	U2	0.1	pDop	-	Position DOP Mask to use
16	U2	0.1	tDop	-	Time DOP Mask to use
18	U2	-	pAcc	m	Position Accuracy Mask
20	U2	-	tAcc	m	Time Accuracy Mask
22	U1	-	staticHoldThresh	cm/s	Static hold threshold
23	U1	-	dgnssTimeout	s	DGNSS timeout
24	U1	-	cnoThreshNumSVs	-	Number of satellites required to have C/N0 above cnoThresh for a fix to be attempted
25	U1	-	cnoThresh	dBHz	C/N0 threshold for deciding whether to attempt a fix
26	U1[2]	-	reserved1	-	Reserved
28	U2	-	staticHoldMaxDist	m	Static hold distance threshold (before quitting static hold)
30	U1	-	utcStandard	-	UTC standard to be used: 0: Automatic; receiver selects based on GNSS configuration (see GNSS time bases). 3: UTC as operated by the U.S. Naval Observatory (USNO); derived from GPS time 6: UTC as operated by the former Soviet Union; derived from GLONASS time 7: UTC as operated by the National Time Service Center, China; derived from BeiDou time
31	U1[5]	-	reserved2	-	Reserved

Bitfield mask

This graphic explains the bits of mask



Name	Description
dyn	Apply dynamic model settings
minEl	Apply minimum elevation settings
posFixMode	Apply fix mode settings
drLim	Reserved
posMask	Apply position mask settings
timeMask	Apply time mask settings
staticHoldMask	Apply static hold settings
dgpsMask	Apply DGPS settings.
cnoThreshold	Apply CNO threshold settings (cnoThresh, cnoThreshNumSVs).
utc	Apply UTC settings.

25.10.14 UBX-CFG-NAVX5 (0x06 0x23)

25.10.14.1 Navigation Engine Expert Settings

Message	**UBX-CFG-NAVX5**					
Description	**Navigation Engine Expert Settings**					
Type	Get/Set					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x23	40	see below	CK_A CK_B

Payload Contents:

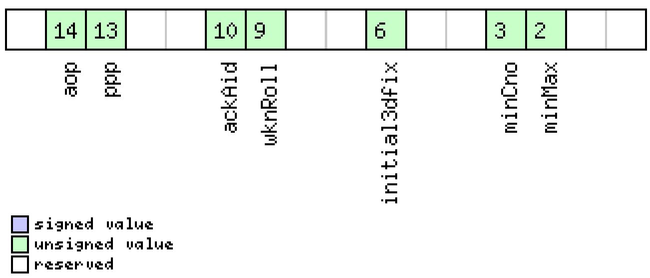
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2	-	version	-	Message version (2 for this version)
2	X2	-	mask1	-	First parameters bitmask. Only the flagged parameters will be applied, unused bits must be set to 0. (see graphic below)
4	X4	-	mask2	-	Second parameters bitmask. Only the flagged parameters will be applied, unused bits must be set to 0. (see graphic below)
8	U1[2]	-	reserved1	-	Reserved
10	U1	-	minSVs	#SVs	Minimum number of satellites for navigation

UBX-CFG-NAVX5 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
11	U1	-	maxSVs	#SVs	Maximum number of satellites for navigation
12	U1	-	minCNO	dBHz	Minimum satellite signal level for navigation
13	U1	-	reserved2	-	Reserved
14	U1	-	iniFix3D	-	1 = initial fix must be 3D
15	U1[2]	-	reserved3	-	Reserved
17	U1	-	ackAiding	-	1 = issue acknowledgements for assistance message input
18	U2	-	wknRollover	-	GPS week rollover number; GPS week numbers will be set correctly from this week up to 1024 weeks after this week. Setting this to 0 reverts to firmware default.
20	U1	-	sigAttenCompMode	dBHz	Only supported on certain products
21	U1	-	reserved4	-	Reserved
22	U1[2]	-	reserved5	-	Reserved
24	U1[2]	-	reserved6	-	Reserved
26	U1	-	usePPP	-	1 = use Precise Point Positioning (only available with the PPP product variant)
27	U1	-	aopCfg	-	AssistNow Autonomous configuration (see graphic below)
28	U1[2]	-	reserved7	-	Reserved
30	U2	-	aopOrbMaxErr	m	Maximum acceptable (modeled) AssistNow Autonomous orbit error (valid range = 5..1000, or 0 = reset to firmware default)
32	U1[4]	-	reserved8	-	Reserved
36	U1[3]	-	reserved9	-	Reserved
39	U1	-	useAdr	-	Only supported on certain products

Bitfield mask1

This graphic explains the bits of mask1



Name	Description
minMax	1 = apply min/max SVs settings
minCno	1 = apply minimum C/N0 setting
initial3dfix	1 = apply initial 3D fix settings
wknRoll	1 = apply GPS weeknumber rollover settings
ackAid	1 = apply assistance acknowledgement settings
ppp	1 = apply usePPP flag
aop	1 = apply aopCfg (useAOP flag) and aopOrbMaxErr settings (AssistNow Autonomous)

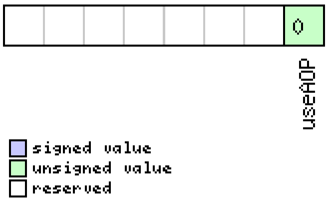
Bitfield mask2

This graphic explains the bits of mask2

Name	Description
adr	Apply ADR/UDR sensor fusion on/off setting (useAdr flag)
sigAttenComp	Only supported on certain products

Bitfield aopCfg

This graphic explains the bits of aopCfg



Name	Description
useAOP	1 = enable AssistNow Autonomous

25.10.15 UBX-CFG-NMEA (0x06 0x17)

25.10.15.1 NMEA protocol configuration (deprecated)

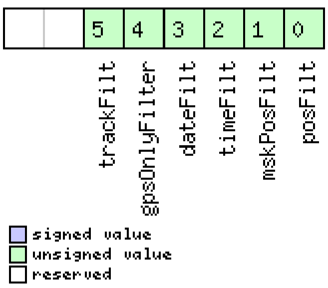
Message	**UBX-CFG-NMEA**					
Description	**NMEA protocol configuration (deprecated)**					
Type	Get/Set					
Comment	**This message version is provided for backwards compatibility only. Use the last version listed below instead (its fields are backwards compatible with this version, it just has extra fields defined).** Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x17	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	filter	-	filter flags (see graphic below)
1	U1	-	nmeaVersion	-	0x23: NMEA version 2.3 0x21: NMEA version 2.1
2	U1	-	numSV	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs
3	X1	-	flags	-	flags (see graphic below)

Bitfield filter

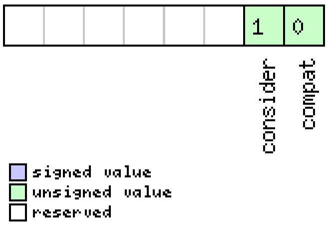
This graphic explains the bits of filter



Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

This graphic explains the bits of flags



Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.

25.10.15.2 NMEA protocol configuration V0 (deprecated)

Message	**UBX-CFG-NMEA**					
Description	**NMEA protocol configuration V0 (deprecated)**					
Type	Get/Set					
Comment	**This message version is provided for backwards compatibility only. Use the last version listed below instead (its fields are backwards compatible with this version, it just has extra fields defined).** Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x17	12	see below	CK_A CK_B

Payload Contents:

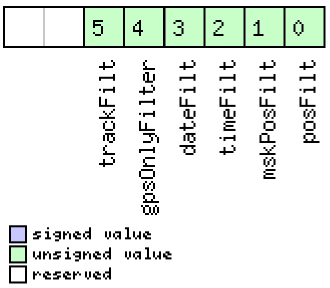
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	filter	-	filter flags (see graphic below)
1	U1	-	nmeaVersion	-	0x23: NMEA version 2.3 0x21: NMEA version 2.1
2	U1	-	numSV	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs

UBX-CFG-NMEA continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
3	X1	-	flags	-	flags (see graphic below)
4	X4	-	gnssToFilter	-	Filters out satellites based on their GNSS. If a bitfield is enabled, the corresponding satellites will be not output. (see graphic below)
8	U1	-	svNumbering	-	Configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID. 0: Strict - Satellites are not output 1: Extended - Use proprietary numbering (see Satellite Numbering)
9	U1	-	mainTalkerId	-	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see UBX-CFG-GNSS). This field enables the main Talker ID to be overridden. 0: Main Talker ID is not overridden 1: Set main Talker ID to 'GP' 2: Set main Talker ID to 'GL' 3: Set main Talker ID to 'GN' 4: Set main Talker ID to 'GA' 5: Set main Talker ID to 'GB'
10	U1	-	gsvTalkerId	-	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden. 0: Use GNSS specific Talker ID (as defined by NMEA) 1: Use the main Talker ID
11	U1	-	version	-	Message version (set to 0 for this version)

Bitfield filter

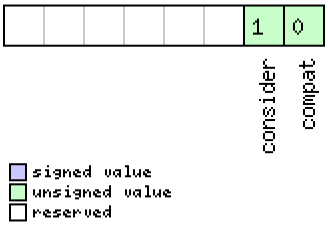
This graphic explains the bits of `filter`



Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

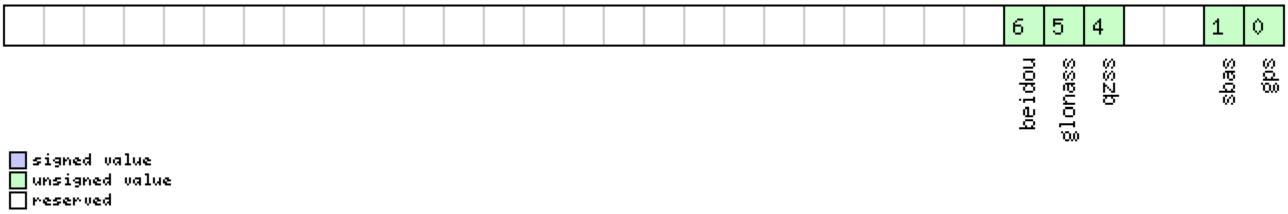
This graphic explains the bits of `flags`



Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.

Bitfield gnssToFilter

This graphic explains the bits of `gnssToFilter`



Name	Description
gps	Disable reporting of GPS satellites
sbas	Disable reporting of SBAS satellites
qzss	Disable reporting of QZSS satellites
glonass	Disable reporting of GLONASS satellites
beidou	Disable reporting of BeiDou satellites

25.10.15.3 Extended NMEA protocol configuration V1

Message	**UBX-CFG-NMEA**					
Description	**Extended NMEA protocol configuration V1**					
Type	Get/Set					
Comment	Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x17	20	see below	CK_A CK_B

Payload Contents:

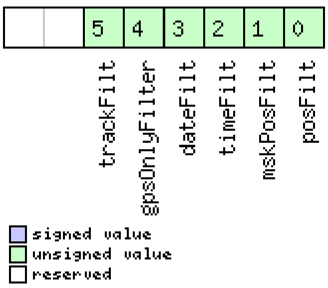
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	`filter`	-	filter flags (see graphic below)
1	U1	-	`nmeaVersion`	-	0x41: NMEA version 4.10 0x40: NMEA version 4.0 0x23: NMEA version 2.3 0x21: NMEA version 2.1
2	U1	-	`numSV`	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs
3	X1	-	`flags`	-	flags (see graphic below)
4	X4	-	`gnssToFilter`	-	Filters out satellites based on their GNSS. If a bitfield is enabled, the corresponding satellites will be not output. (see graphic below)
8	U1	-	`svNumbering`	-	Configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID. 0: Strict - Satellites are not output 1: Extended - Use proprietary numbering (see Satellite Numbering)

UBX-CFG-NMEA continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
9	U1	-	mainTalkerId	-	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see UBX-CFG-GNSS). This field enables the main Talker ID to be overridden. 0: Main Talker ID is not overridden 1: Set main Talker ID to 'GP' 2: Set main Talker ID to 'GL' 3: Set main Talker ID to 'GN' 4: Set main Talker ID to 'GA' 5: Set main Talker ID to 'GB'
10	U1	-	gsvTalkerId	-	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden. 0: Use GNSS specific Talker ID (as defined by NMEA) 1: Use the main Talker ID
11	U1	-	version	-	Message version (set to 1 for this version)
12	CH[2]	-	bdsTalkerId	-	Sets the two characters that should be used for the BeiDou Talker ID If these are set to zero, the default BeiDou TalkerId will be used
14	U1[6]	-	reserved1	-	Reserved

Bitfield filter

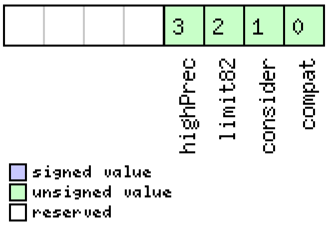
This graphic explains the bits of filter



Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

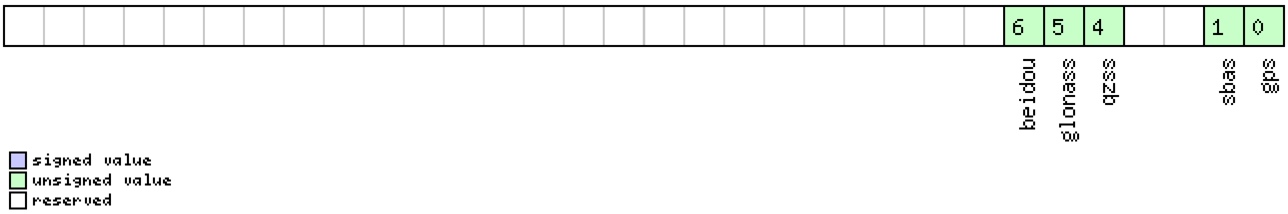
This graphic explains the bits of flags



Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.
limit82	enable strict limit to 82 characters maximum.
highPrec	enable high precision mode. This flag cannot be set in conjunction with either Compatibility Mode or Limit82 Mode.

Bitfield gnssToFilter

This graphic explains the bits of gnssToFilter



Name	Description
gps	Disable reporting of GPS satellites
sbas	Disable reporting of SBAS satellites
qzss	Disable reporting of QZSS satellites
glonass	Disable reporting of GLONASS satellites
beidou	Disable reporting of BeiDou satellites

25.10.16 UBX-CFG-ODO (0x06 0x1E)

25.10.16.1 Odometer, Low-speed COG Engine Settings

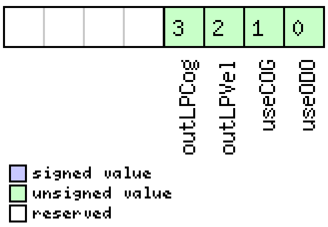
Message	**UBX-CFG-ODO**					
Description	**Odometer, Low-speed COG Engine Settings**					
Type	Get/Set					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1E	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U1	-	flags	-	Odometer/Low-speed COG filter flags (see graphic below)
5	X1	-	odoCfg	-	Odometer filter settings (see graphic below)
6	U1[6]	-	reserved2	-	Reserved
12	U1	1e-1	cogMaxSpeed	m/s	Speed below which course-over-ground (COG) is computed with the low-speed COG filter
13	U1	-	cogMaxPosAcc	m	Maximum acceptable position accuracy for computing COG with the low-speed COG filter
14	U1[2]	-	reserved3	-	Reserved
16	U1	-	velLpGain	-	Velocity low-pass filter level, range 0..255
17	U1	-	cogLpGain	-	COG low-pass filter level (at speed < 8 m/s), range 0..255
18	U1[2]	-	reserved4	-	Reserved

Bitfield flags

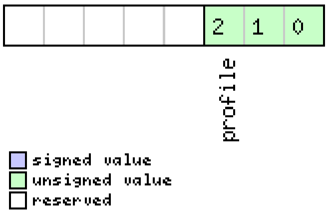
This graphic explains the bits of flags



Name	Description
`useODO`	Odometer enabled flag
`useCOG`	Low-speed COG filter enabled flag
`outLPVel`	Output low-pass filtered velocity flag
`outLPCog`	Output low-pass filtered heading (COG) flag

Bitfield odoCfg

This graphic explains the bits of `odoCfg`



Name	Description
`profile`	Profile type (0=running, 1=cycling, 2=swimming, 3=car, 4=custom)

25.10.17 UBX-CFG-PM2 (0x06 0x3B)

25.10.17.1 Extended Power Management configuration

Message	**UBX-CFG-PM2**					
Description	**Extended Power Management configuration**					
Type	Get/Set					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x3B	48	see below	CK_A CK_B

Payload Contents:

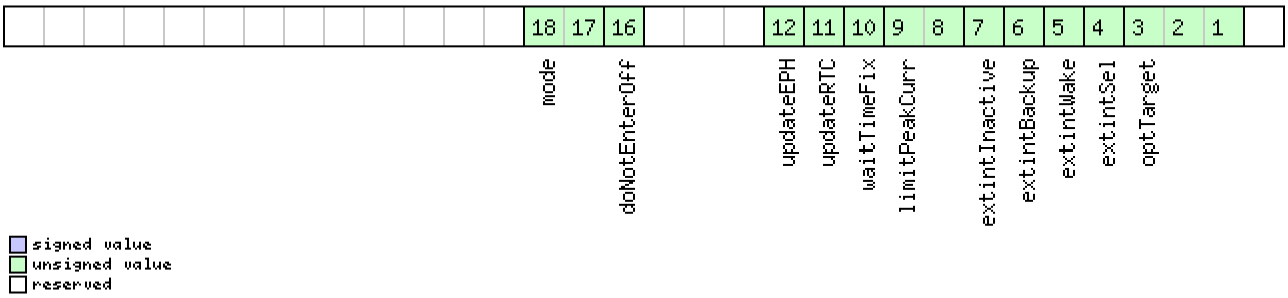
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (0x02 for this version) Note: the message version number is the same as for protocol versions 18 up to 22; please select correct message version based on the protocol version supported by your firmware.
1	U1	-	`reserved1`	-	Reserved
2	U1	-	`maxStartupStateDur`	s	Maximum time to spend in Acquisition state. If 0: bound disabled (see maxStartupStateDur). (not supported in protocol versions 23 to 23.01)
3	U1	-	`reserved2`	-	Reserved
4	X4	-	`flags`	-	PSM configuration flags (see graphic below)
8	U4	-	`updatePeriod`	ms	Position update period. If set to 0, the receiver will never retry a fix and it will wait for external events

UBX-CFG-PM2 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
12	U4	-	searchPeriod	ms	Acquisition retry period if previously failed. If set to 0, the receiver will never retry a startup (not supported in protocol versions 23 to 23.01)
16	U4	-	gridOffset	ms	Grid offset relative to GPS start of week (not supported in protocol versions 23 to 23.01)
20	U2	-	onTime	s	Time to stay in Tracking state (not supported in protocol versions 23 to 23.01)
22	U2	-	minAcqTime	s	minimal search time
24	U1[20]	-	reserved3	-	Reserved
44	U4	-	extintInactivityMs	ms	inactivity time out on EXTINT pint if enabled (not supported in protocol versions 23 to 23.01).

Bitfield flags

This graphic explains the bits of flags



Name	Description
optTarget	Optimization Target 000 performance (default) 001 power save 010 reserved 011 reserved 100 reserved 101 reserved 110 reserved 111 reserved
extintSel	EXTINT Pin Select 0 EXTINT0 1 EXTINT1
extintWake	EXTINT Pin Control 0 disabled 1 enabled, keep receiver awake as long as selected EXTINT pin is 'high' (not supported in protocol versions 23 to 23.01)

Bitfield flags Description continued

Name	Description
extintBackup	EXTINT Pin Control 0 disabled 1 enabled, force receiver into BACKUP mode when selected EXTINT pin is 'low'
extintInactive	EXTINT Pin Control 0 disabled 1 enabled, Force backup in case EXTINT Pin is inactive for time longer than extintIncactivityMs (not supported in protocol versions 23 to 23.01)
limitPeakCurr	Limit Peak Current 00 disabled 01 enabled, peak current is limited 10 reserved 11 reserved
waitTimeFix	Wait for Timefix (see waitTimeFix) 0 wait for normal fix ok before starting on time 1 wait for time fix ok before starting on time (not supported in protocol versions 23 to 23.01)
updateRTC	Update Real Time Clock (see updateRTC) 0 Do not wake up to update RTC. RTC is updated during normal on-time. 1 Update RTC. The receiver adds extra wake-up cycles to update the RTC. (not supported in protocol versions 23 to 23.01)
updateEPH	Update Ephemeris (see updateEPH) 0 Do not wake up to update Ephemeris data 1 Update Ephemeris. The receiver adds extra wake-up cycles to update the Ephemeris data
doNotEnterOff	Behavior of receiver in case of no fix (see doNotEnterOff) 0 receiver enters (Inactive) Awaiting Next Search state 1 receiver does not enter (Inactive) Awaiting Next Search state but keeps trying to acquire a fix instead (not supported in protocol versions 23 to 23.01)
mode	Mode of operation (see mode) 00 ON/OFF operation (PSMOO) (not supported in protocol versions 23 to 23.01) 01 Cyclic tracking operation (PSMCT) 10 reserved 11 reserved

25.10.18 UBX-CFG-PMS (0x06 0x86)

25.10.18.1 Power Mode Setup

Message	**UBX-CFG-PMS**				
Description	**Power Mode Setup**				
Type	Get/Set				
Comment	Using UBX-CFG-PMS to set Super-E mode 1, 2, 4Hz navigation rates sets 180 s minAcqTime instead the default 300 s in protocol version 23.01.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x86	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x00 for this version)
1	U1	-	powerSetupValue	-	Power setup value 0x00 -> Full power 0x01 -> Balanced 0x02 -> Interval 0x03 -> Aggressive with 1Hz 0x04 -> Aggressive with 2Hz 0x05 -> Aggressive with 4Hz 0xFF -> Invalid (only when polling)
2	U2	-	period	s	Position update period and search period. Recommended minimum period is 10s, although the receiver accepts any value bigger than 5s. Only valid when powerSetupValueset to Interval, otherwise must be set to '0'.
4	U2	-	onTime	s	Duration of the ON phase, must be smaller than the period. Only valid when powerSetupValue set to Interval, otherwise must be set to '0'.
6	U1[2]	-	reserved1	-	Reserved

25.10.19 UBX-CFG-PRT (0x06 0x00)

25.10.19.1 Polls the configuration for one I/O Port

Message	**UBX-CFG-PRT**					
Description	**Polls the configuration for one I/O Port**					
Type	Poll Request					
Comment	Sending this message with a port ID as payload results in having the receiver return the configuration for the specified port.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	PortID	-	Port Identifier Number (see the other versions of CFG-PRT for valid values)

25.10.19.2 Port Configuration for UART

Message	**UBX-CFG-PRT**					
Description	**Port Configuration for UART**					
Type	Get/Set					
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit. Note that this message can affect baud rate and other transmission parameters. Because there may be messages queued for transmission there may be uncertainty about which protocol applies to such messages. In addition a message currently in transmission may be corrupted by a protocol change. Host data reception parameters may have to be changed to be able to receive future messages, including the acknowledge message resulting from the CFG-PRT message.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B

Payload Contents:

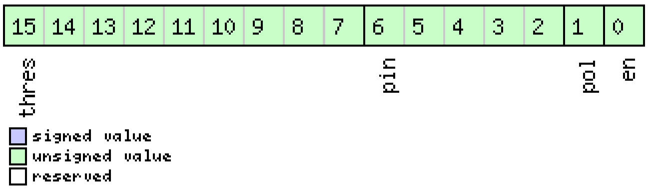
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	portID	-	Port Identifier Number (see Integration Manual for valid UART port IDs)
1	U1	-	reserved1	-	Reserved
2	X2	-	txReady	-	TX ready PIN configuration (see graphic below)
4	X4	-	mode	-	A bit mask describing the UART mode (see graphic below)
8	U4	-	baudRate	Bits/s	Baud rate in bits/second

UBX-CFG-PRT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
12	X2	-	inProtoMask	-	A mask describing which input protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
14	X2	-	outProtoMask	-	A mask describing which output protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
16	X2	-	flags	-	Flags bit mask (see graphic below)
18	U1[2]	-	reserved2	-	Reserved

Bitfield txReady

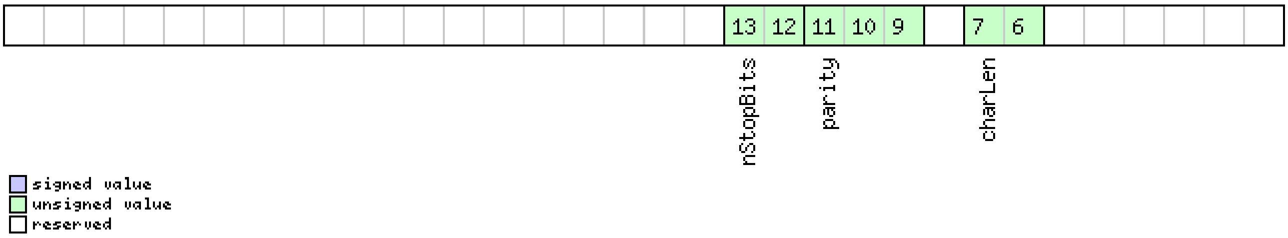
This graphic explains the bits of txReady



Name	Description
en	Enable TX ready feature for this port
pol	Polarity 0 High-active 1 Low-active
pin	PIO to be used (must not be in use already by another function)
thres	Threshold The given threshold is multiplied by 8 bytes. The TX ready PIN goes active after >= thres*8 bytes are pending for the port and going inactive after the last pending bytes have been written to hardware (0-4 bytes before end of stream). 0x000 no threshold 0x001 8byte 0x002 16byte ... 0x1FE 4080byte 0x1FF 4088byte

Bitfield mode

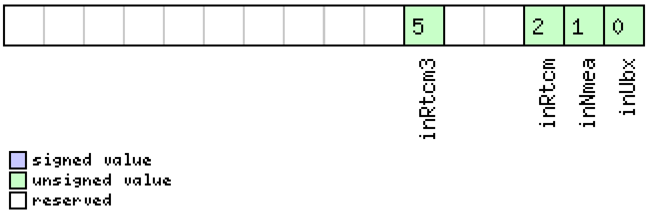
This graphic explains the bits of `mode`



Name	Description
`charLen`	Character Length 00 5bit (not supported) 01 6bit (not supported) 10 7bit (supported only with parity) 11 8bit
`parity`	000 Even Parity 001 Odd Parity 10X No Parity X1X Reserved
`nStopBits`	Number of Stop Bits 00 1 Stop Bit 01 1.5 Stop Bit 10 2 Stop Bit 11 0.5 Stop Bit

Bitfield inProtoMask

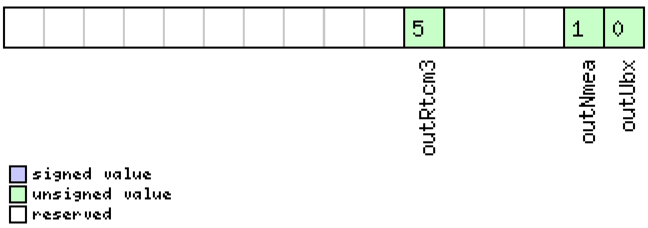
This graphic explains the bits of `inProtoMask`



Name	Description
`inUbx`	UBX protocol
`inNmea`	NMEA protocol
`inRtcm`	RTCM2 protocol
`inRtcm3`	RTCM3 protocol

Bitfield outProtoMask

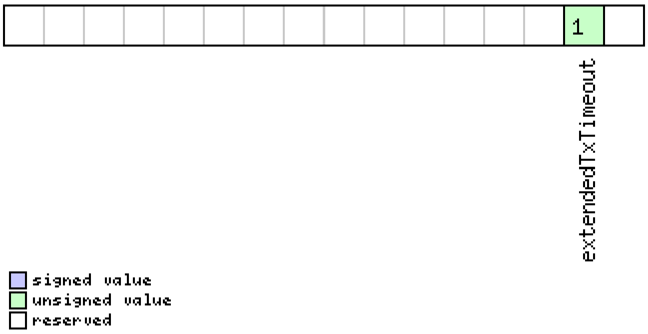
This graphic explains the bits of `outProtoMask`



Name	Description
outUbx	UBX protocol
outNmea	NMEA protocol
outRtcm3	RTCM3 protocol

Bitfield flags

This graphic explains the bits of flags



Name	Description
extendedTxTimeout	Extended TX timeout: if set, the port will timeout if allocated TX memory >=4 kB and no activity for 1.5s. If not set the port will timeout if no activity for 1.5s regardless on the amount of allocated TX memory.

25.10.19.3 Port Configuration for SPI Port

Message	**UBX-CFG-PRT**					
Description	**Port Configuration for SPI Port**					
Type	Get/Set					
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B

Payload Contents:

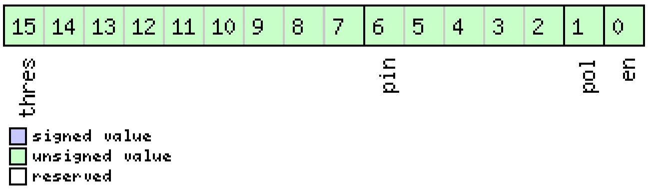
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	portID	-	Port Identifier Number (= 4 for SPI port)
1	U1	-	reserved1	-	Reserved
2	X2	-	txReady	-	TX ready PIN configuration (see graphic below)
4	X4	-	mode	-	SPI Mode Flags (see graphic below)
8	U1[4]	-	reserved2	-	Reserved
12	X2	-	inProtoMask	-	A mask describing which input protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)

UBX-CFG-PRT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
14	X2	-	outProtoMask	-	A mask describing which output protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
16	X2	-	flags	-	Flags bit mask (see graphic below)
18	U1[2]	-	reserved3	-	Reserved

Bitfield txReady

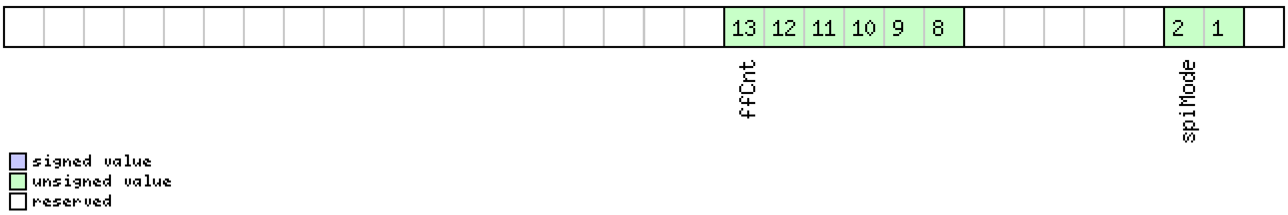
This graphic explains the bits of txReady



Name	Description
en	Enable TX ready feature for this port
pol	Polarity 0 High-active 1 Low-active
pin	PIO to be used (must not be in use already by another function)
thres	Threshold The given threshold is multiplied by 8 bytes. The TX ready PIN goes active after >= thres*8 bytes are pending for the port and going inactive after the last pending bytes have been written to hardware (0-4 bytes before end of stream). 0x000 no threshold 0x001 8byte 0x002 16byte ... 0x1FE 4080byte 0x1FF 4088byte

Bitfield mode

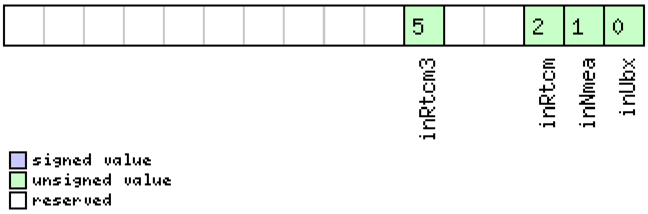
This graphic explains the bits of mode



Name	Description
spiMode	00 SPI Mode 0: CPOL = 0, CPHA = 0 01 SPI Mode 1: CPOL = 0, CPHA = 1 10 SPI Mode 2: CPOL = 1, CPHA = 0 11 SPI Mode 3: CPOL = 1, CPHA = 1
ffCnt	Number of bytes containing 0xFF to receive before switching off reception. Range: 0(mechanism off)-63

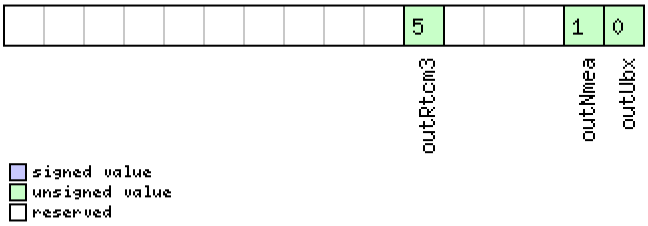
Bitfield inProtoMask

This graphic explains the bits of `inProtoMask`



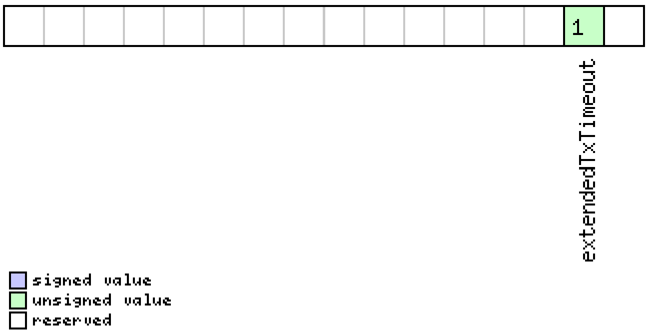
Bitfield outProtoMask

This graphic explains the bits of `outProtoMask`



Bitfield flags

This graphic explains the bits of `flags`



Name	Description
extendedTxTimeout	Extended TX timeout: if set, the port will timeout if allocated TX memory >=4 kB and no activity for 1.5s.

25.10.19.4 Port Configuration for DDC Port

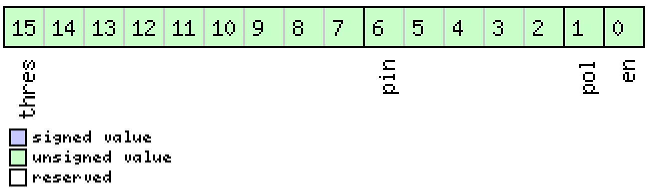
Message	**UBX-CFG-PRT**					
Description	**Port Configuration for DDC Port**					
Type	Get/Set					
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	portID	-	Port Identifier Number (= 0 for DDC port)
1	U1	-	reserved1	-	Reserved
2	X2	-	txReady	-	TX ready PIN configuration (see graphic below)
4	X4	-	mode	-	DDC Mode Flags (see graphic below)
8	U1[4]	-	reserved2	-	Reserved
12	X2	-	inProtoMask	-	A mask describing which input protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
14	X2	-	outProtoMask	-	A mask describing which output protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
16	X2	-	flags	-	Flags bit mask (see graphic below)
18	U1[2]	-	reserved3	-	Reserved

Bitfield txReady

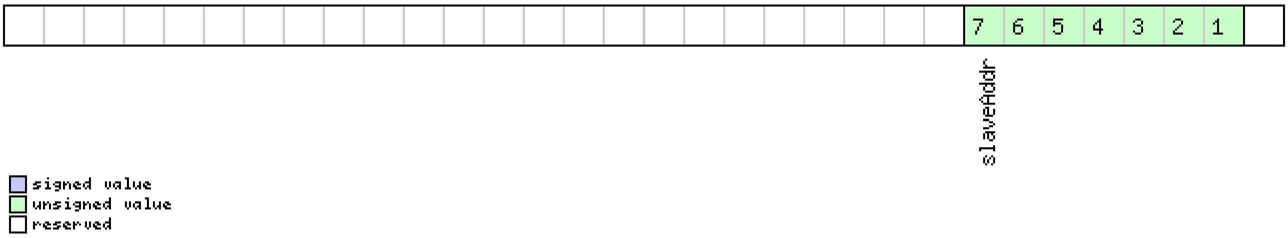
This graphic explains the bits of txReady



Name	Description
en	Enable TX ready feature for this port
pol	Polarity 0 High-active 1 Low-active
pin	PIO to be used (must not be in use already by another function)
thres	Threshold The given threshold is multiplied by 8 bytes. The TX ready PIN goes active after >= thres*8 bytes are pending for the port and going inactive after the last pending bytes have been written to hardware (0-4 bytes before end of stream). 0x000 no threshold 0x001 8byte 0x002 16byte ... 0x1FE 4080byte 0x1FF 4088byte

Bitfield mode

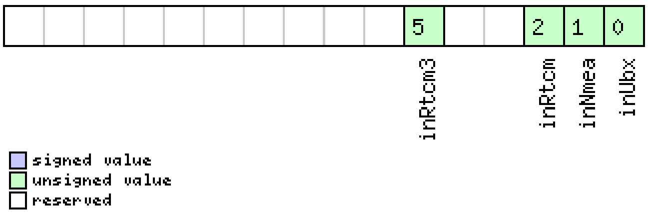
This graphic explains the bits of mode



Name	Description
slaveAddr	Slave address Range: 0x07 < slaveAddr < 0x78. Bit 0 must be 0

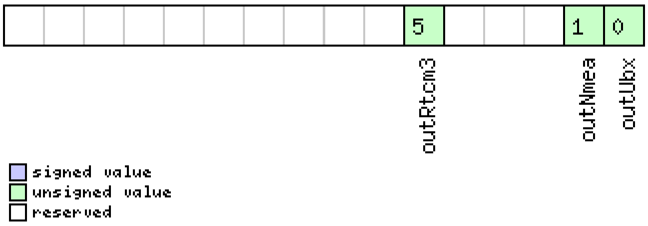
Bitfield inProtoMask

This graphic explains the bits of inProtoMask



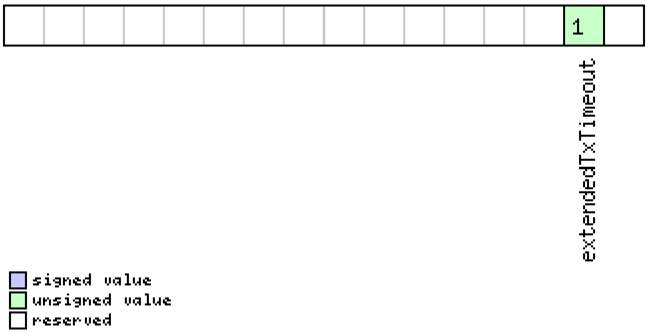
Bitfield outProtoMask

This graphic explains the bits of `outProtoMask`



Bitfield flags

This graphic explains the bits of `flags`



Name	Description
`extendedTxTimeout`	Extended TX timeout: if set, the port will timeout if allocated TX memory >=4 kB and no activity for 1.5s.

25.10.20 UBX-CFG-PWR (0x06 0x57)

25.10.20.1 Put receiver in a defined power state.

Message	**UBX-CFG-PWR**					
Description	**Put receiver in a defined power state.**					
Type	Set					
Comment	**This message is deprecated in protocol versions greater than 17. Use `UBX-CFG-RST` for GNSS start/stop and `UBX-RXM-PMREQ` for software backup.** -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x57	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (1 for this version)
1	U1[3]	-	`reserved1`	-	Reserved

UBX-CFG-PWR continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	U4	-	state	-	Enter system state 0x52554E20: GNSS running 0x53544F50: GNSS stopped 0x42434B50: Software Backup. USB interface will be disabled, other wakeup source is needed.

25.10.21 UBX-CFG-RATE (0x06 0x08)

25.10.21.1 Navigation/Measurement Rate Settings

Message	**UBX-CFG-RATE**					
Description	**Navigation/Measurement Rate Settings**					
Type	Get/Set					
Comment	This message allows the user to alter the rate at which navigation solutions (and the measurements that they depend on) are generated by the receiver. The calculation of the navigation solution will always be aligned to the top of a second zero (first second of the week) of the configured reference time system. (Navigation period is an integer multiple of the measurement period in protocol versions greater than 17) • Each measurement triggers the measurements generation and raw data output. • The navRate value defines that every nth measurement triggers a navigation epoch. • The update rate has a direct influence on the power consumption. The more fixes that are required, the more CPU power and communication resources are required. • For most applications a 1 Hz update rate would be sufficient. • When using Power Save Mode, measurement and navigation rate can differ from the values configured here. • See Measurement and navigation rate with Power Save Mode for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x08	6	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2	-	measRate	ms	The elapsed time between GNSS measurements, which defines the rate, e.g. 100ms => 10Hz, 1000ms => 1Hz, 10000ms => 0.1Hz. Measurement rate should be greater than or equal to 25 ms. (Measurement rate should be greater than or equal to 50 ms in protocol versions less than 24)

UBX-CFG-RATE continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U2	-	navRate	cycles	The ratio between the number of measurements and the number of navigation solutions, e.g. 5 means five measurements for every navigation solution. Maximum value is 127.
4	U2	-	timeRef	-	The time system to which measurements are aligned: 0: UTC time 1: GPS time 2: GLONASS time 3: BeiDou time 4: Galileo time

25.10.22 UBX-CFG-RINV (0x06 0x34)

25.10.22.1 Contents of Remote Inventory

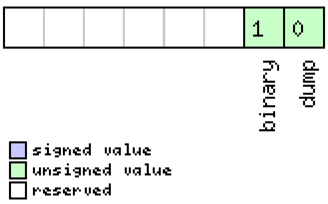
Message	**UBX-CFG-RINV**					
Description	**Contents of Remote Inventory**					
Type	Get/Set					
Comment	If N is greater than 30, the excess bytes are discarded.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x34	1 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	flags	-	Flags (see graphic below)
Start of repeated block (N times)					
1 + 1*N	U1	-	data	-	Data to store/stored in Remote Inventory.
End of repeated block					

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
dump	Dump data at startup. Does not work if flag binary is set.
binary	Data is binary.

25.10.23 UBX-CFG-RST (0x06 0x04)

25.10.23.1 Reset Receiver / Clear Backup Data Structures

Message	**UBX-CFG-RST**
Description	**Reset Receiver / Clear Backup Data Structures**
Type	Command
Comment	Don't expect this message to be acknowledged by the receiver. • Newer FW version won't acknowledge this message at all. • Older FW version will acknowledge this message but the acknowledge may not be sent completely before the receiver is reset.

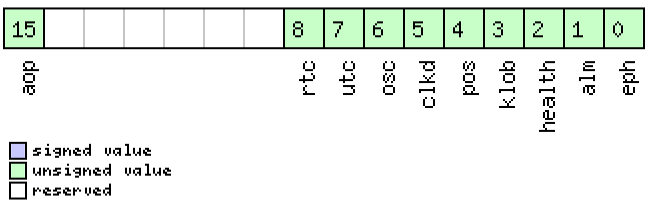
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x06	0x04	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X2	-	navBbrMask	-	BBR Sections to clear. The following Special Sets apply: 0x0000 Hot start 0x0001 Warm start 0xFFFF Cold start (see graphic below)
2	U1	-	resetMode	-	Reset Type 0x00 - Hardware reset (Watchdog) immediately 0x01 - Controlled Software reset 0x02 - Controlled Software reset (GNSS only) 0x04 - Hardware reset (Watchdog) after shutdown 0x08 - Controlled GNSS stop 0x09 - Controlled GNSS start
3	U1	-	reserved1	-	Reserved

Bitfield navBbrMask

This graphic explains the bits of navBbrMask



Name	Description
eph	Ephemeris
alm	Almanac
health	Health
klob	Klobuchar parameters
pos	Position
clkd	Clock Drift
osc	Oscillator Parameter
utc	UTC Correction + GPS Leap Seconds Parameters
rtc	RTC
aop	Autonomous Orbit Parameters

25.10.24 UBX-CFG-RXM (0x06 0x11)

25.10.24.1 RXM configuration

Message	**UBX-CFG-RXM**				
Description	**RXM configuration**				
Type	Get/Set				
Comment	For a detailed description see section Power Management.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x11	2	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	reserved1	-	Reserved
1	U1	-	lpMode	-	Low Power Mode 0: Continuous Mode 1: Power Save Mode 4: Continuous Mode

25.10.25 UBX-CFG-SBAS (0x06 0x16)

25.10.25.1 SBAS Configuration

Message	**UBX-CFG-SBAS**				
Description	**SBAS Configuration**				
Type	Get/Set				
Comment	This message configures the SBAS receiver subsystem (i.e. WAAS, EGNOS, MSAS). See the SBAS Configuration Settings Description for a detailed description of how these settings affect receiver operation.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x16	8	see below	CK_A CK_B

Payload Contents:

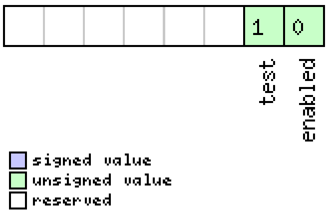
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	mode	-	SBAS Mode (see graphic below)
1	X1	-	usage	-	SBAS Usage (see graphic below)

UBX-CFG-SBAS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U1	-	maxSBAS	-	Maximum Number of SBAS prioritized tracking channels (valid range: 0 - 3) to use (obsolete and superseded by UBX-CFG-GNSS in protocol versions 14+).
3	X1	-	scanmode2	-	Continuation of scanmode bitmask below (see graphic below)
4	X4	-	scanmode1	-	Which SBAS PRN numbers to search for (Bitmask) If all Bits are set to zero, auto-scan (i.e. all valid PRNs) are searched. Every bit corresponds to a PRN number (see graphic below)

Bitfield mode

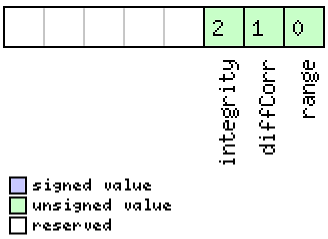
This graphic explains the bits of mode



Name	Description
enabled	SBAS Enabled (1) / Disabled (0) - This field is deprecated; use UBX-CFG-GNSS to enable/disable SBAS operation
test	SBAS Testbed: Use data anyhow (1) / Ignore data when in Test Mode (SBAS Msg 0)

Bitfield usage

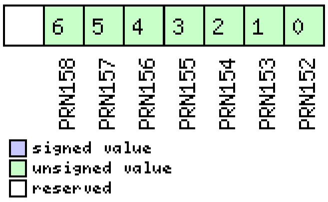
This graphic explains the bits of usage



Name	Description
range	Use SBAS GEOs as a ranging source (for navigation)
diffCorr	Use SBAS Differential Corrections
integrity	Use SBAS Integrity Information

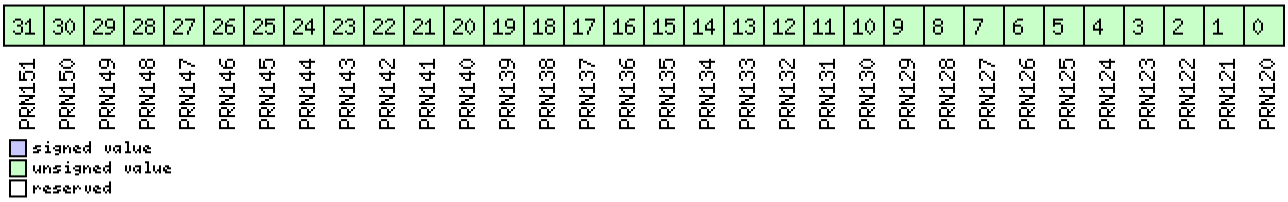
Bitfield scanmode2

This graphic explains the bits of scanmode2



Bitfield scanmode1

This graphic explains the bits of scanmode1



25.10.26 UBX-CFG-USB (0x06 0x1B)

25.10.26.1 USB Configuration

Message	**UBX-CFG-USB**					
Description	**USB Configuration**					
Type	Get/Set					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1B	108	see below	CK_A CK_B

Payload Contents:

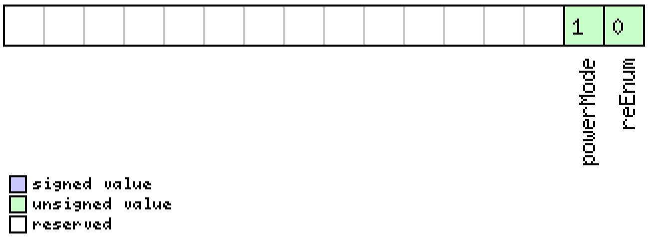
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2	-	vendorID	-	Vendor ID. This field shall only be set to registered Vendor IDs. Changing this field requires special Host drivers.
2	U2	-	productID	-	Product ID. Changing this field requires special Host drivers.
4	U1[2]	-	reserved1	-	Reserved
6	U1[2]	-	reserved2	-	Reserved
8	U2	-	powerConsumption	mA	Power consumed by the device
10	X2	-	flags	-	various configuration flags (see graphic below)

UBX-CFG-USB continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
12	CH[32]	-	vendorString	-	String containing the vendor name. 32 ASCII bytes including 0-termination.
44	CH[32]	-	productString	-	String containing the product name. 32 ASCII bytes including 0-termination.
76	CH[32]	-	serialNumber	-	String containing the serial number. 32 ASCII bytes including 0-termination. Changing the String fields requires special Host drivers.

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
reEnum	force re-enumeration
powerMode	self-powered (1), bus-powered (0)

25.11 UBX-INF (0x04)

Information Messages: i.e. Printf-Style Messages, with IDs such as Error, Warning, Notice.
Messages in the INF class are used to output strings in a printf style from the firmware or application code. All INF messages have an associated type to indicate the kind of message.

25.11.1 UBX-INF-DEBUG (0x04 0x04)

25.11.1.1 ASCII output with debug contents

Message	**UBX-INF-DEBUG**					
Description	**ASCII output with debug contents**					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x04	0 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*1	CH	-	str	-	ASCII Character
End of repeated block					

25.11.2 UBX-INF-ERROR (0x04 0x00)

25.11.2.1 ASCII output with error contents

Message	**UBX-INF-ERROR**					
Description	**ASCII output with error contents**					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x00	0 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*1	CH	-	str	-	ASCII Character
End of repeated block					

25.11.3 UBX-INF-NOTICE (0x04 0x02)

25.11.3.1 ASCII output with informational contents

Message	**UBX-INF-NOTICE**				
Description	**ASCII output with informational contents**				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x02	0 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*1	CH	-	str	-	ASCII Character
End of repeated block					

25.11.4 UBX-INF-TEST (0x04 0x03)

25.11.4.1 ASCII output with test contents

Message	**UBX-INF-TEST**				
Description	**ASCII output with test contents**				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x03	0 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*1	CH	-	str	-	ASCII Character
End of repeated block					

25.11.5 UBX-INF-WARNING (0x04 0x01)

25.11.5.1 ASCII output with warning contents

Message	**UBX-INF-WARNING**				
Description	**ASCII output with warning contents**				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x01	0 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					
N*1	CH	-	str	-	ASCII Character

UBX-INF-WARNING continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
End of repeated block					

25.12 UBX-LOG (0x21)

Logging Messages: i.e. Log creation, deletion, info and retrieval.
Messages in the LOG class are used to configure and report status information of the logging and batching features.

25.12.1 UBX-LOG-BATCH (0x21 0x11)

25.12.1.1 Batched data

Message	**UBX-LOG-BATCH**					
Description	**Batched data**					
Type	Polled					
Comment	**Note that during a leap second there may be more (or less) than 60 seconds in a minute; see the description of leap seconds for details.** This message combines position, velocity and time solution, including accuracy figures. The output of this message can be requested via `UBX-LOG-RETRIEVEBATCH`. The content of this message is influenced by `UBX-CFG-BATCH`. Depending on the flags `extraPvt` and `extraOdo` some of the fields in this message may not be valid. This validity information is also indicated in this message via flags of the same name. See Data Batching for more information.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x11	100	see below	CK_A CK_B

Payload Contents:

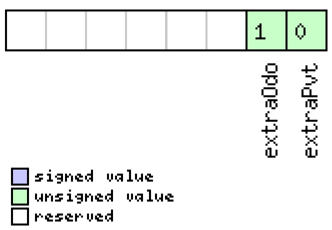
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (0x00 for this version)
1	X1	-	`contentValid`	-	Content validity flags (see graphic below)
2	U2	-	`msgCnt`	-	Message counter; increments for each sent UBX-LOG-BATCH message.
4	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details. Only valid if `extraPvt` is set.
8	U2	-	`year`	y	Year (UTC)
10	U1	-	`month`	month	Month, range 1..12 (UTC)
11	U1	-	`day`	d	Day of month, range 1..31 (UTC)
12	U1	-	`hour`	h	Hour of day, range 0..23 (UTC)
13	U1	-	`min`	min	Minute of hour, range 0..59 (UTC)
14	U1	-	`sec`	s	Seconds of minute, range 0..60 (UTC)
15	X1	-	`valid`	-	Validity flags (see graphic below)
16	U4	-	`tAcc`	ns	Time accuracy estimate (UTC) Only valid if `extraPvt` is set.
20	I4	-	`fracSec`	ns	Fraction of second, range -1e9 .. 1e9 (UTC)
24	U1	-	`fixType`	-	GNSSfix Type: 0: no fix 2: 2D-fix 3: 3D-fix

UBX-LOG-BATCH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
25	X1	-	flags	-	Fix status flags (see graphic below)
26	X1	-	flags2	-	Additional flags
27	U1	-	numSV	-	Number of satellites used in Nav Solution Only valid if extraPvt is set.
28	I4	1e-7	lon	deg	Longitude
32	I4	1e-7	lat	deg	Latitude
36	I4	-	height	mm	Height above ellipsoid
40	I4	-	hMSL	mm	Height above mean sea level Only valid if extraPvt is set.
44	U4	-	hAcc	mm	Horizontal accuracy estimate
48	U4	-	vAcc	mm	Vertical accuracy estimate Only valid if extraPvt is set.
52	I4	-	velN	mm/s	NED north velocity Only valid if extraPvt is set.
56	I4	-	velE	mm/s	NED east velocity Only valid if extraPvt is set.
60	I4	-	velD	mm/s	NED down velocity Only valid if extraPvt is set.
64	I4	-	gSpeed	mm/s	Ground Speed (2-D)
68	I4	1e-5	headMot	deg	Heading of motion (2-D)
72	U4	-	sAcc	mm/s	Speed accuracy estimate Only valid if extraPvt is set.
76	U4	1e-5	headAcc	deg	Heading accuracy estimate Only valid if extraPvt is set.
80	U2	0.01	pDOP	-	Position DOP Only valid if extraPvt is set.
82	U1[2]	-	reserved1	-	Reserved
84	U4	-	distance	m	Ground distance since last reset Only valid if extraOdo is set.
88	U4	-	totalDistance	m	Total cumulative ground distance Only valid if extraOdo is set.
92	U4	-	distanceStd	m	Ground distance accuracy (1-sigma) Only valid if extraOdo is set.
96	U1[4]	-	reserved2	-	Reserved

Bitfield contentValid

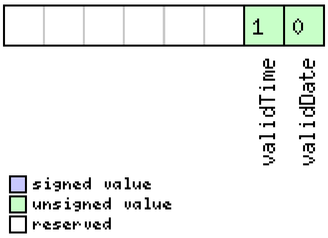
This graphic explains the bits of contentValid



Name	Description
`extraPvt`	Store extra PVT information The fields `iTOW`, `tAcc`, `numSV`, `hMSL`, `vAcc`, `velN`, `velE`, `velD`, `sAcc`, `headAcc` and `pDOP` are only valid if this flag is set.
`extraOdo`	Store odometer data The fields `distance`, `totalDistance` and `distanceStd` are only valid if this flag is set. Note: the odometer feature itself must also be enabled.

Bitfield valid

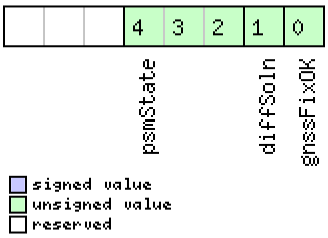
This graphic explains the bits of `valid`



Name	Description
`validDate`	1 = valid UTC Date (see Time Validity section for details)
`validTime`	1 = valid UTC Time of Day (see Time Validity section for details)

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
`gnssFixOK`	1 = valid fix (i.e within DOP & accuracy masks)
`diffSoln`	1 = differential corrections were applied
`psmState`	Power Save Mode state (see Power Management): 0: PSM is not active 1: Enabled (an intermediate state before Acquisition state 2: Acquisition 3: Tracking 4: Power Optimized Tracking 5: Inactive

25.12.2 UBX-LOG-CREATE (0x21 0x07)

25.12.2.1 Create Log File

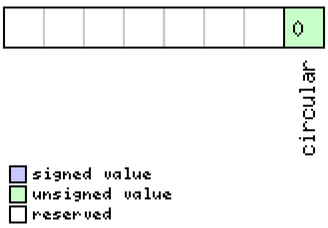
Message	**UBX-LOG-CREATE**					
Description	**Create Log File**					
Type	Command					
Comment	This message is used to create an initial logging file and activate the logging subsystem. UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure. This message does not handle activation of recording or filtering of log entries (see UBX-CFG-LOGFILTER).					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x07	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	The version of this message. Set to 0
1	X1	-	logCfg	-	Config flags (see graphic below)
2	U1	-	reserved1	-	Reserved
3	U1	-	logSize	-	Indicates the size of the log: 0 (maximum safe size): Ensures that logging will not be interrupted and enough space will be left available for all other uses of the filestore 1 (minimum size): 2 (user defined): See 'userDefinedSize' below
4	U4	-	userDefinedSize	bytes	Sets the maximum amount of space in the filestore that can be used by the logging task. This field is only applicable if logSize is set to user defined.

Bitfield logCfg

This graphic explains the bits of logCfg



Name	Description
circular	Log is circular (new entries overwrite old ones in a full log) if this bit set

25.12.3 UBX-LOG-ERASE (0x21 0x03)

25.12.3.1 Erase Logged Data

Message	**UBX-LOG-ERASE**					
Description	**Erase Logged Data**					
Type	Command					
Comment	This message deactivates the logging system and erases all logged data. UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x03	0	see below	CK_A CK_B
No payload						

25.12.4 UBX-LOG-FINDTIME (0x21 0x0E)

25.12.4.1 Find index of a log entry based on a given time

Message	**UBX-LOG-FINDTIME**					
Description	**Find index of a log entry based on a given time**					
Type	Input					
Comment	This message can be used for a time-based search of a log. It can find the index of the first log entry with time equal to the given time, otherwise the index of the most recent entry with time less than the given time. This index can then be used with the UBX-LOG-RETRIEVE message to provide time-based retrieval of log entries. Searching a log is effective for a given time later than the base date (January 1st, 2004). Searching a log for a given time earlier than the base date will result in an 'entry not found' response. Searching a log for a given time greater than the last recorded entry's time will return the index of the last recorded entry.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x0E	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (=0 for this version)
1	U1	-	type	-	Message type, 0 for request
2	U1[2]	-	reserved1	-	Reserved
4	U2	-	year	-	Year (1-65635) of UTC time
6	U1	-	month	-	Month (1-12) of UTC time
7	U1	-	day	-	Day (1-31) of UTC time
8	U1	-	hour	-	Hour (0-23) of UTC time
9	U1	-	minute	-	Minute (0-59) of UTC time
10	U1	-	second	-	Second (0-60) of UTC time
11	U1	-	reserved2	-	Reserved

25.12.4.2 Response to FINDTIME request

Message	**UBX-LOG-FINDTIME**					
Description	**Response to FINDTIME request**					
Type	Output					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0E	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (=1 for this version)
1	U1	-	type	-	Message type, 1 for response
2	U1[2]	-	reserved1	-	Reserved
4	U4	-	entryNumber	-	Index of the first log entry with time = given time, otherwise index of the most recent entry with time < given time. If 0xFFFFFFFF, no log entry found with time <= given time. The indexing of log entries is zero based.

25.12.5 UBX-LOG-INFO (0x21 0x08)

25.12.5.1 Poll for log information

Message	**UBX-LOG-INFO**					
Description	**Poll for log information**					
Type	Poll Request					
Comment	Upon sending of this message, the receiver returns UBX-LOG-INFO as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x08	0	see below	CK_A CK_B

No payload

25.12.5.2 Log information

Message	**UBX-LOG-INFO**
Description	**Log information**
Type	Output
Comment	This message is used to report information about the logging subsystem. Note: • The reported maximum log size will be smaller than that originally specified in LOG-CREATE due to logging and filestore implementation overheads. • Log entries are compressed in a variable length fashion, so it may be difficult to predict log space usage with any precision. • There may be times when the receiver does not have an accurate time (e.g. if the week number is not yet known), in which case some entries will not have a timestamp. This may result in the oldest/newest entry time values not taking account of these entries.

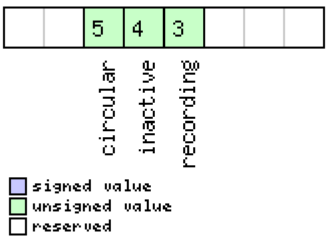
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x08	48	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	The version of this message. Set to 1
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	filestoreCapacity	bytes	The capacity of the filestore
8	U1[8]	-	reserved2	-	Reserved
16	U4	-	currentMaxLogSize	bytes	The maximum size the current log is allowed to grow to
20	U4	-	currentLogSize	bytes	Approximate amount of space in log currently occupied
24	U4	-	entryCount	-	Number of entries in the log. Note: for circular logs this value will decrease when a group of entries is deleted to make space for new ones.
28	U2	-	oldestYear	-	Oldest entry UTC year (1-65635) or zero if there are no entries with known time
30	U1	-	oldestMonth	-	Oldest month (1-12)
31	U1	-	oldestDay	-	Oldest day (1-31)
32	U1	-	oldestHour	-	Oldest hour (0-23)
33	U1	-	oldestMinute	-	Oldest minute (0-59)
34	U1	-	oldestSecond	-	Oldest second (0-60)
35	U1	-	reserved3	-	Reserved
36	U2	-	newestYear	-	Newest year (1-65635) or zero if there are no entries with known time
38	U1	-	newestMonth	-	Newest month (1-12)
39	U1	-	newestDay	-	Newest day (1-31)
40	U1	-	newestHour	-	Newest hour (0-23)
41	U1	-	newestMinute	-	Newest minute (0-59)
42	U1	-	newestSecond	-	Newest second (0-60)
43	U1	-	reserved4	-	Reserved
44	X1	-	status	-	Log status flags (see graphic below)
45	U1[3]	-	reserved5	-	Reserved

Bitfield status

This graphic explains the bits of `status`



Name	Description
recording	Log entry recording is currently turned on
inactive	Logging system not active - no log present
circular	The current log is circular

25.12.6 UBX-LOG-RETRIEVEBATCH (0x21 0x10)

25.12.6.1 Request batch data

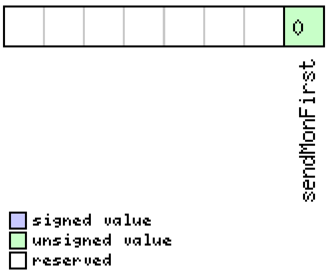
Message	**UBX-LOG-RETRIEVEBATCH**					
Description	**Request batch data**					
Type	Command					
Comment	This message is used to request batched data. Batch entries are returned in chronological order, using one UBX-LOG-BATCH per navigation epoch. The speed of transfer can be maximized by using a high data rate. See Data Batching for more information.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x10	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x00 for this version)
1	X1	-	flags	-	Flags (see graphic below)
2	U1[2]	-	reserved1	-	Reserved

Bitfield flags

This graphic explains the bits of flags



Name	Description
sendMonFirst	Send UBX-MON-BATCH message before sending the UBX-LOG-BATCH message(s).

25.12.7 UBX-LOG-RETRIEVEPOSEXTRA (0x21 0x0f)

25.12.7.1 Odometer log entry

Message	**UBX-LOG-RETRIEVEPOSEXTRA**				
Description	**Odometer log entry**				
Type	Output				
Comment	This message is used to report an odometer log entry				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0f	32	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	entryIndex	-	The index of this log entry
4	U1	-	version	-	The version of this message. Set to 0
5	U1	-	reserved1	-	Reserved
6	U2	-	year	-	Year (1-65635) of UTC time. Will be zero if time not known
8	U1	-	month	-	Month (1-12) of UTC time
9	U1	-	day	-	Day (1-31) of UTC time
10	U1	-	hour	-	Hour (0-23) of UTC time
11	U1	-	minute	-	Minute (0-59) of UTC time
12	U1	-	second	-	Second (0-60) of UTC time
13	U1[3]	-	reserved2	-	Reserved
16	U4	-	distance	-	Odometer distance traveled since the last time the odometer was reset by a UBX-NAV-RESETODO
20	U1[12]	-	reserved3	-	Reserved

25.12.8 UBX-LOG-RETRIEVEPOS (0x21 0x0b)

25.12.8.1 Position fix log entry

Message	**UBX-LOG-RETRIEVEPOS**				
Description	**Position fix log entry**				
Type	Output				
Comment	This message is used to report a position fix log entry				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0b	40	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	entryIndex	-	The index of this log entry
4	I4	1e-7	lon	deg	Longitude
8	I4	1e-7	lat	deg	Latitude
12	I4	-	hMSL	mm	Height above mean sea level
16	U4	-	hAcc	mm	Horizontal accuracy estimate
20	U4	-	gSpeed	mm/s	Ground speed (2-D)
24	U4	1e-5	heading	deg	Heading

UBX-LOG-RETRIEVEPOS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
28	U1	-	version	-	The version of this message. Set to 0
29	U1	-	fixType	-	Fix type: 0x01: Dead Reckoning only 0x02: 2D-Fix 0x03: 3D-Fix 0x04: GNSS + Dead Reckoning combined
30	U2	-	year	-	Year (1-65635) of UTC time
32	U1	-	month	-	Month (1-12) of UTC time
33	U1	-	day	-	Day (1-31) of UTC time
34	U1	-	hour	-	Hour (0-23) of UTC time
35	U1	-	minute	-	Minute (0-59) of UTC time
36	U1	-	second	-	Second (0-60) of UTC time
37	U1	-	reserved1	-	Reserved
38	U1	-	numSV	-	Number of satellites used in the position fix
39	U1	-	reserved2	-	Reserved

25.12.9 UBX-LOG-RETRIEVESTRING (0x21 0x0d)

25.12.9.1 Byte string log entry

Message	**UBX-LOG-RETRIEVESTRING**					
Description	**Byte string log entry**					
Type	Output					
Comment	This message is used to report a byte string log entry					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x0d	16 + 1*byteCount	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	entryIndex	-	The index of this log entry
4	U1	-	version	-	The version of this message. Set to 0
5	U1	-	reserved1	-	Reserved
6	U2	-	year	-	Year (1-65635) of UTC time. Will be zero if time not known
8	U1	-	month	-	Month (1-12) of UTC time
9	U1	-	day	-	Day (1-31) of UTC time
10	U1	-	hour	-	Hour (0-23) of UTC time
11	U1	-	minute	-	Minute (0-59) of UTC time
12	U1	-	second	-	Second (0-60) of UTC time
13	U1	-	reserved2	-	Reserved
14	U2	-	byteCount	-	Size of string in bytes
Start of repeated block (byteCount times)					
16 + 1*N	U1	-	bytes	-	The bytes of the string
End of repeated block					

25.12.10 UBX-LOG-RETRIEVE (0x21 0x09)

25.12.10.1 Request log data

Message	UBX-LOG-RETRIEVE					
Description	**Request log data**					
Type	Command					
Comment	This message is used to request logged data (log recording must first be disabled, see UBX-CFG-LOGFILTER). Log entries are returned in chronological order, using the messages UBX-LOG-RETRIEVEPOS and UBX-LOG-RETRIEVESTRING. If the odometer was enabled at the time a position was logged, then message UBX-LOG-RETRIEVEPOSEXTRA will also be used. The maximum number of entries that can be returned in response to a single UBX-LOG-RETRIEVE message is 256. If more entries than this are required the message will need to be sent multiple times with different startNumbers. The retrieve will be stopped if any UBX-LOG message is received. The speed of transfer can be maximized by using a high data rate and temporarily stopping the GPS processing (see UBX-CFG-RST).					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x09	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`startNumber`	-	Index of first log entry to be transferred. If it is larger than the index of the last available log entry, then the first log entry to be transferred is the last available log entry. The indexing of log entries is zero based.
4	U4	-	`entryCount`	-	Number of log entries to transfer in total including the first entry to be transferred. If it is larger than the log entries available starting from the first entry to be transferred, then only the available log entries are transferred followed by a UBX-ACK-NAK. The maximum is 256.
8	U1	-	`version`	-	The version of this message. Set to 0.
9	U1[3]	-	`reserved1`	-	Reserved

25.12.11 UBX-LOG-STRING (0x21 0x04)

25.12.11.1 Store arbitrary string in on-board flash

Message	**UBX-LOG-STRING**					
Description	**Store arbitrary string in on-board flash**					
Type	Command					
Comment	This message can be used to store an arbitrary byte string in the on-board flash memory. The maximum length that can be stored is 256 bytes.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x21	0x04	0 + 1*N	see below	CK_A CK_B

Byte Offset	Number Format	Scaling	Name	Unit	Description
Payload Contents:					
Start of repeated block (N times)					
N*1	U1	-	bytes	-	The string of bytes to be logged (maximum 256)
End of repeated block					

25.13 UBX-MGA (0x13)

Multiple GNSS Assistance Messages: i.e. Assistance data for various GNSS.
Messages in the MGA class are used for GNSS aiding information from and to the receiver.

25.13.1 UBX-MGA-ACK (0x13 0x60)

25.13.1.1 UBX-MGA-ACK-DATA0

Message	**UBX-MGA-ACK-DATA0**					
Description	**Multiple GNSS Acknowledge message**					
Type	Output					
Comment	This message is sent by a u-blox receiver to acknowledge the receipt of an assistance message. Acknowledgments are enabled by setting the ackAiding parameter in the UBX-CFG-NAVX5 message. See the description of flow control for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x60	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Type of acknowledgment: 0: The message was not used by the receiver (see infoCode field for an indication of why) 1: The message was accepted for use by the receiver (the infoCode field will be 0)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	infoCode	-	Provides greater information on what the receiver chose to do with the message contents: 0: The receiver accepted the data 1: The receiver doesn't know the time so can't use the data (To resolve this a UBX-MGA-INI-TIME_UTC message should be supplied first) 2: The message version is not supported by the receiver 3: The message size does not match the message version 4: The message data could not be stored to the database 5: The receiver is not ready to use the message data 6: The message type is unknown
3	U1	-	msgId	-	UBX message ID of the ack'ed message
4	U1[4]	-	msgPayloadStart	-	The first 4 bytes of the ack'ed message's payload

25.13.2 UBX-MGA-ANO (0x13 0x20)

25.13.2.1 Multiple GNSS AssistNow Offline Assistance

Message	**UBX-MGA-ANO**					
Description	**Multiple GNSS AssistNow Offline Assistance**					
Type	Input					
Comment	This message is created by the AssistNow Offline service to deliver AssistNow Offline assistance to the receiver. See the description of AssistNow Offline for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x20	76	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x00 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	Satellite identifier (see Satellite Numbering)
3	U1	-	gnssId	-	GNSS identifier (see Satellite Numbering)
4	U1	-	year	-	years since the year 2000
5	U1	-	month	-	month (1..12)
6	U1	-	day	-	day (1..31)
7	U1	-	reserved1	-	Reserved
8	U1[64]	-	data	-	assistance data
72	U1[4]	-	reserved2	-	Reserved

25.13.3 UBX-MGA-BDS (0x13 0x03)

25.13.3.1 UBX-MGA-BDS-EPH

Message	**UBX-MGA-BDS-EPH**					
Description	**BDS Ephemeris Assistance**					
Type	Input					
Comment	This message allows the delivery of BeiDou ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x03	88	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	BDS satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1	-	SatH1	-	Autonomous satellite Health flag
5	U1	-	IODC	-	Issue of Data, Clock
6	I2	2^-66	a2	s/s^2	Time polynomial coefficient 2

UBX-MGA-BDS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
8	I4	2^-50	a1	s/s	Time polynomial coefficient 1
12	I4	2^-33	a0	s	Time polynomial coefficient 0
16	U4	2^3	toc	s	Clock data reference time
20	I2	0.1	TGD1	ns	Equipment Group Delay Differential
22	U1	-	URAI	-	User Range Accuracy Index
23	U1	-	IODE	-	Issue of Data, Ephemeris
24	U4	2^3	toe	s	Ephemeris reference time
28	U4	2^-19	sqrtA	m^0.5	Square root of semi-major axis
32	U4	2^-33	e	-	Eccentricity
36	I4	2^-31	omega	semi-circles	Argument of perigee
40	I2	2^-43	Deltan	semi-circles /s	Mean motion difference from computed value
42	I2	2^-43	IDOT	semi-circles /s	Rate of inclination angle
44	I4	2^-31	M0	semi-circles	Mean anomaly at reference time
48	I4	2^-31	Omega0	semi-circles	Longitude of ascending node of orbital of plane computed according to reference time
52	I4	2^-43	OmegaDot	semi-circles /s	Rate of right ascension
56	I4	2^-31	i0	semi-circles	Inclination angle at reference time
60	I4	2^-31	Cuc	semi-circles	Amplitude of cosine harmonic correction term to the argument of latitude
64	I4	2^-31	Cus	semi-circles	Amplitude of sine harmonic correction term to the argument of latitude
68	I4	2^-6	Crc	m	Amplitude of cosine harmonic correction term to the orbit radius
72	I4	2^-6	Crs	m	Amplitude of sine harmonic correction term to the orbit radius
76	I4	2^-31	Cic	semi-circles	Amplitude of cosine harmonic correction term to the angle of inclination
80	I4	2^-31	Cis	semi-circles	Amplitude of sine harmonic correction term to the angle of inclination
84	U1[4]	-	reserved2	-	Reserved

25.13.3.2 UBX-MGA-BDS-ALM

Message	**UBX-MGA-BDS-ALM**					
Description	**BDS Almanac Assistance**					
Type	Input					
Comment	This message allows the delivery of BeiDou almanac assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x03	40	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this version)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	BeiDou satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1	-	Wna	week	Almanac Week Number
5	U1	2^12	toa	s	Almanac reference time
6	I2	2^-19	deltaI	semi-circles	Almanac correction of orbit reference inclination at reference time
8	U4	2^-11	sqrtA	m^0.5	Almanac square root of semi-major axis
12	U4	2^-21	e	-	Almanac eccentricity
16	I4	2^-23	omega	semi-circles	Almanac argument of perigee
20	I4	2^-23	M0	semi-circles	Almanac mean anomaly at reference time
24	I4	2^-23	Omega0	semi-circles	Almanac longitude of ascending node of orbit plane at computed according to reference time
28	I4	2^-38	omegaDot	semi-circles /s	Almanac rate of right ascension
32	I2	2^-20	a0	s	Almanac satellite clock bias
34	I2	2^-38	a1	s/s	Almanac satellite clock rate
36	U1[4]	-	reserved2	-	Reserved

25.13.3.3 UBX-MGA-BDS-HEALTH

Message	**UBX-MGA-BDS-HEALTH**					
Description	**BDS Health Assistance**					
Type	Input					
Comment	This message allows the delivery of BeiDou health assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x03	68	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x04 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	U2[30]	-	healthCode	-	Each two-byte value represents a BDS SV (1-30). The 9 LSBs of each byte contain the 9 bit health code from subframe 5 pages 7,8 of the D1 message, and from subframe 5 pages 35,36 of the D1 message.
64	U1[4]	-	reserved2	-	Reserved

25.13.3.4 UBX-MGA-BDS-UTC

Message	**UBX-MGA-BDS-UTC**					
Description	**BDS UTC Assistance**					
Type	Input					
Comment	This message allows the delivery of BeiDou UTC assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x03	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x05 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I4	2^-30	a0UTC	s	BDT clock bias relative to UTC
8	I4	2^-50	a1UTC	s/s	BDT clock rate relative to UTC
12	I1	-	dtLS	s	Delta time due to leap seconds before the new leap second effective
13	U1[1]	-	reserved2	-	Reserved
14	U1	-	wnRec	week	BeiDou week number of reception of this UTC parameter set (8 bit truncated)
15	U1	-	wnLSF	week	Week number of the new leap second
16	U1	-	dN	day	Day number of the new leap second

UBX-MGA-BDS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
17	I1	-	dtLSF	s	Delta time due to leap seconds after the new leap second effective
18	U1[2]	-	reserved3	-	Reserved

25.13.3.5 UBX-MGA-BDS-IONO

Message	**UBX-MGA-BDS-IONO**					
Description	**BDS Ionospheric Assistance**					
Type	Input					
Comment	This message allows the delivery of BeiDou ionospheric assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x03	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x06 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I1	2^-30	alpha0	s	Ionospheric parameter alpha0
5	I1	2^-27	alpha1	s/pi	Ionospheric parameter alpha1
6	I1	2^-24	alpha2	s/pi^2	Ionospheric parameter alpha2
7	I1	2^-24	alpha3	s/pi^3	Ionospheric parameter alpha3
8	I1	2^11	beta0	s	Ionospheric parameter beta0
9	I1	2^14	beta1	s/pi	Ionospheric parameter beta1
10	I1	2^16	beta2	s/pi^2	Ionospheric parameter beta2
11	I1	2^16	beta3	s/pi^3	Ionospheric parameter beta3
12	U1[4]	-	reserved2	-	Reserved

25.13.4 UBX-MGA-DBD (0x13 0x80)

25.13.4.1 Poll the Navigation Database

Message	**UBX-MGA-DBD**					
Description	**Poll the Navigation Database**					
Type	Poll Request					
Comment	Poll the whole navigation data base. The receiver will send all available data from its internal database. The receiver will indicate the finish of the transmission with a UBX-MGA-ACK. The msgPayloadStart field of the UBX-MGA-ACK message will contain a U4 representing the number of UBX-MGA-DBD-DATA* messages sent.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x80	0	see below	CK_A CK_B
No payload						

25.13.4.2 Navigation Database Dump Entry

Message	**UBX-MGA-DBD**					
Description	**Navigation Database Dump Entry**					
Type	Input/Output					
Comment	**UBX-MGA-DBD messages are only intended to be sent back to the same receiver that generated them.** Navigation database entry. The data fields are firmware specific. Transmission of this type of message will be acknowledged by UBX-MGA-ACK messages, if acknowledgment has been enabled (see the description of flow control for details). The maximum payload size for firmware 2.01 onwards is 164 bytes (which makes the maximum message size 172 bytes).					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x80	12 + 1*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1[12]	-	reserved1	-	Reserved
Start of repeated block (N times)					
12 + 1*N	U1	-	data	-	fw specific data
End of repeated block					

25.13.5 UBX-MGA-FLASH (0x13 0x21)

25.13.5.1 UBX-MGA-FLASH-DATA

Message	**UBX-MGA-FLASH-DATA**					
Description	**Transfer MGA-ANO data block to flash**					
Type	Input					
Comment	This message is used to transfer a block of MGA-ANO data from host to the receiver. Upon reception of this message, the receiver will write the payload data to its internal non-volatile memory (flash). Also, on reception of the first MGA-FLASH-DATA message, the receiver will erase the flash allocated to storing any existing MGA-ANO data. The payload can be up to 512 bytes. Payloads larger than this would exceed the receiver's internal buffering capabilities. The receiver will ACK/NACK this message using the message alternatives given below. The host shall wait for an acknowledge message before sending the next data block. See Flash-based AssistNow Offline for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x21	6 + 1*size	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)

UBX-MGA-FLASH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U2	-	sequence	-	Message sequence number, starting at 0 and increamenting by 1 for each MGA-FLASH-DATA message sent.
4	U2	-	size	-	Payload size in bytes.
Start of repeated block (size times)					
6 + 1*N	U1	-	data	-	Payload data.
End of repeated block					

25.13.5.2 UBX-MGA-FLASH-STOP

Message	**UBX-MGA-FLASH-STOP**					
Description	**Finish flashing MGA-ANO data**					
Type	Input					
Comment	This message is used to tell the receiver that there are no more MGA-FLASH type 1 messages coming, and that it can do any final internal operations needed to commit the data to flash as a background activity. A UBX-MGA-ACK message will be sent at the end of this process. Note that there may be a delay of several seconds before the UBX-MGA-ACK for this message is sent because of the time taken for this processing. See Flash-based AssistNow Offline for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x21	2	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this type)
1	U1	-	version	-	Message version (0x00 for this version)

25.13.5.3 UBX-MGA-FLASH-ACK

Message	**UBX-MGA-FLASH-ACK**					
Description	**Acknowledge last FLASH-DATA or -STOP**					
Type	Output					
Comment	This message reports an ACK/NACK to the host for the last MGA-FLASH type 1 or type 2 message message received. See Flash-based AssistNow Offline for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x21	6	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x03 for this type)
1	U1	-	version	-	Message version (0x00 for this version)

UBX-MGA-FLASH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U1	-	ack	-	Acknowledgment type. 0 - ACK: Message received and written to flash. 1 - NACK: Problem with last message, re-transmission required (this only happens while acknowledging a UBX-MGA_FLASH_DATA message). 2 - NACK: problem with last message, give up.
3	U1	-	reserved1	-	Reserved
4	U2	-	sequence	-	If acknowledging a UBX-MGA-FLASH-DATA message this is the Message sequence number being ack'ed. If acknowledging a UBX-MGA-FLASH-STOP message it will be set to 0xffff.

25.13.6 UBX-MGA-GAL (0x13 0x02)

25.13.6.1 UBX-MGA-GAL-EPH

Message	UBX-MGA-GAL-EPH					
Description	**Galileo Ephemeris Assistance**					
Type	Input					
Comment	This message allows the delivery of Galileo ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x02	76	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	Galileo Satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U2	-	iodNav	-	Ephemeris and clock correction Issue of Data
6	I2	2^-43	deltaN	semi-circles /s	Mean motion difference from computed value
8	I4	2^-31	m0	semi-circles	Mean anomaly at reference time
12	U4	2^-33	e	-	Eccentricity
16	U4	2^-19	sqrtA	m^0.5	Square root of the semi-major axis
20	I4	2^-31	omega0	semi-circles	Longitude of ascending node of orbital plane at weekly epoch

UBX-MGA-GAL continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
24	I4	2^-31	i0	semi-circles	Inclination angle at reference time
28	I4	2^-31	omega	semi-circles	Argument of perigee
32	I4	2^-43	omegaDot	semi-circles /s	Rate of change of right ascension
36	I2	2^-43	iDot	semi-circles /s	Rate of change of inclination angle
38	I2	2^-29	cuc	radians	Amplitude of the cosine harmonic correction term to the argument of latitude
40	I2	2^-29	cus	radians	Amplitude of the sine harmonic correction term to the argument of latitude
42	I2	2^-5	crc	radians	Amplitude of the cosine harmonic correction term to the orbit radius
44	I2	2^-5	crs	radians	Amplitude of the sine harmonic correction term to the orbit radius
46	I2	2^-29	cic	radians	Amplitude of the cosine harmonic correction term to the angle of inclination
48	I2	2^-29	cis	radians	Amplitude of the sine harmonic correction term to the angle of inclination
50	U2	60	toe	s	Ephemeris reference time
52	I4	2^-34	af0	s	SV clock bias correction coefficient
56	I4	2^-46	af1	s/s	SV clock drift correction coefficient
60	I1	2^-59	af2	s/s squared	SV clock drift rate correction coefficient
61	U1	-	sisaIndexE1E5b	-	Signal-In-Space Accuracy index for dual frequency E1-E5b
62	U2	60	toc	s	Clock correction data reference Time of Week
64	I2	-	bgdE1E5b	-	E1-E5b Broadcast Group Delay
66	U1[2]	-	reserved2	-	Reserved
68	U1	-	healthE1B	-	E1-B Signal Health Status
69	U1	-	dataValidityE1B	-	E1-B Data Validity Status
70	U1	-	healthE5b	-	E5b Signal Health Status
71	U1	-	dataValidityE5b	-	E5b Data Validity Status
72	U1[4]	-	reserved3	-	Reserved

25.13.6.2 UBX-MGA-GAL-ALM

Message	UBX-MGA-GAL-ALM					
Description	**Galileo Almanac Assistance**					
Type	Input					
Comment	This message allows the delivery of Galileo almanac assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x02	32	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	Galileo Satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1	-	ioda	-	Almanac Issue of Data
5	U1	-	almWNa	week	Almanac reference week number
6	U2	600	toa	s	Almanac reference time
8	I2	2^-9	deltaSqrtA	m^0.5	Difference with respect to the square root of the nominal semi-major axis (29 600 km)
10	U2	2^-16	e	-	Eccentricity
12	I2	2^-14	deltaI	semi-circles	Inclination at reference time relative to i0 = 56 degree
14	I2	2^-15	omega0	semi-circles	Longitude of ascending node of orbital plane at weekly epoch
16	I2	2^-33	omegaDot	semi-circles /s	Rate of change of right ascension
18	I2	2^-15	omega	semi-circles	Argument of perigee
20	I2	2^-15	m0	semi-circles	Satellite mean anomaly at reference time
22	I2	2^-19	af0	s	Satellite clock correction bias 'truncated'
24	I2	2^-38	af1	s/s	Satellite clock correction linear 'truncated'
26	U1	-	healthE1B	-	Satellite E1-B signal health status
27	U1	-	healthE5b	-	Satellite E5b signal health status
28	U1[4]	-	reserved2	-	Reserved

25.13.6.3 UBX-MGA-GAL-TIMEOFFSET

Message	**UBX-MGA-GAL-TIMEOFFSET**					
Description	**Galileo GPS time offset assistance**					
Type	Input					
Comment	This message allows the delivery of Galileo time to GPS time offset. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x02	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x03 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I2	2^-35	a0G	s	Constant term of the polynomial describing the offset
6	I2	2^-51	a1G	s/s	Rate of change of the offset
8	U1	3600	t0G	s	DReference time for GGTO data
9	U1	-	wn0G	weeks	Week Number of GGTO reference
10	U1[2]	-	reserved2	-	Reserved

25.13.6.4 UBX-MGA-GAL-UTC

Message	**UBX-MGA-GAL-UTC**					
Description	**Galileo UTC Assistance**					
Type	Input					
Comment	This message allows the delivery of Galileo UTC assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x02	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x05 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I4	2^-30	a0	s	First parameter of UTC polynomial
8	I4	2^-50	a1	s/s	Second parameter of UTC polynomial
12	I1	-	dtLS	s	Delta time due to current leap seconds
13	U1	3600	tot	s	UTC parameters reference time of week (Galileo time)
14	U1	-	wnt	weeks	UTC parameters reference week number (the 8 bit WNt field)
15	U1	-	wnLSF	weeks	Week number at the end of which the future leap second becomes effective (the 8 bit WNLSF field)

UBX-MGA-GAL continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16	U1	-	dN	days	Day number at the end of which the future leap second becomes effective
17	I1	-	dTLSF	s	Delta time due to future leap seconds
18	U1[2]	-	reserved2	-	Reserved

25.13.7 UBX-MGA-GLO (0x13 0x06)

25.13.7.1 UBX-MGA-GLO-EPH

Message	**UBX-MGA-GLO-EPH**				
Description	**GLONASS Ephemeris Assistance**				
Type	Input				
Comment	This message allows the delivery of GLONASS ephemeris assistance to a receiver. See the description of AssistNow Online for details.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x06	48	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	GLONASS Satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1	-	FT	-	User range accuracy
5	U1	-	B	-	Health flag from string 2
6	U1	-	M	-	Type of GLONASS satellite (1 indicates GLONASS-M)
7	I1	-	H	-	Carrier frequency number of navigation RF signal, Range=(-7 .. 6), -128 for unknown
8	I4	2^-11	x	km	X component of the SV position in PZ-90. 02 coordinate System
12	I4	2^-11	y	km	Y component of the SV position in PZ-90. 02 coordinate System
16	I4	2^-11	z	km	Z component of the SV position in PZ-90. 02 coordinate System
20	I4	2^-20	dx	km/s	X component of the SV velocity in PZ-90. 02 coordinate System
24	I4	2^-20	dy	km/s	Y component of the SV velocity in PZ-90. 02 coordinate System
28	I4	2^-20	dz	km/s	Z component of the SV velocity in PZ-90. 02 coordinate System
32	I1	2^-30	ddx	km/s^2	X component of the SV acceleration in PZ-90.02 coordinate System

UBX-MGA-GLO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
33	I1	2^-30	ddy	km/s^2	Y component of the SV acceleration in PZ-90.02 coordinate System
34	I1	2^-30	ddz	km/s^2	Z component of the SV acceleration in PZ-90.02 coordinate System
35	U1	15	tb	minutes	Index of a time interval within current day according to UTC(SU)
36	I2	2^-40	gamma	-	Relative carrier frequency deviation
38	U1	-	E	days	Ephemeris data age indicator
39	I1	2^-30	deltaTau	s	Time difference between L2 and L1 band
40	I4	2^-30	tau	s	SV clock bias
44	U1[4]	-	reserved2	-	Reserved

25.13.7.2 UBX-MGA-GLO-ALM

Message	**UBX-MGA-GLO-ALM**				
Description	**GLONASS Almanac Assistance**				
Type	Input				
Comment	This message allows the delivery of GLONASS almanac assistance to a receiver. See the description of AssistNow Online for details.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x06	36	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	GLONASS Satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U2	-	N	days	Reference calender day number of almanac within the four-year period (from string 5)
6	U1	-	M	-	Type of GLONASS satellite (1 indicates GLONASS-M)
7	U1	-	C	-	Unhealthy flag at instant of almanac upload (1 indicates operability of satellite)
8	I2	2^-18	tau	s	Coarse time correction to GLONASS time
10	U2	2^-20	epsilon	-	Eccentricity
12	I4	2^-20	lambda	semi-circles	Longitude of the first (within the N-day) ascending node of satellite orbit in PC-90.02 coordinate system
16	I4	2^-20	deltaI	semi-circles	Correction to the mean value of inclination
20	U4	2^-5	tLambda	s	Time of the first ascending node passage

UBX-MGA-GLO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
24	I4	2^-9	deltaT	s/orbital-period	Correction to the mean value of Draconian period
28	I1	2^-14	deltaDT	s/orbital-period^2	Rate of change of Draconian period
29	I1	-	H	-	Carrier frequency number of navigation RF signal, Range=(-7 .. 6)
30	I2	-	omega	-	Argument of perigee
32	U1[4]	-	reserved2	-	Reserved

25.13.7.3 UBX-MGA-GLO-TIMEOFFSET

Message	**UBX-MGA-GLO-TIMEOFFSET**					
Description	**GLONASS Auxiliary Time Offset Assistance**					
Type	Input					
Comment	This message allows the delivery of auxiliary GLONASS assistance (including the GLONASS time offsets to other GNSS systems) to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x06	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x03 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U2	-	N	days	Reference calendar day number within the four-year period of almanac (from string 5)
4	I4	2^-27	tauC	s	Time scale correction to UTC(SU) time
8	I4	2^-31	tauGps	s	Correction to GPS time relative to GLONASS time
12	I2	2^-10	B1	s	Coefficient to determine delta UT1
14	I2	2^-16	B2	s/msd	Rate of change of delta UT1
16	U1[4]	-	reserved1	-	Reserved

25.13.8 UBX-MGA-GPS (0x13 0x00)

25.13.8.1 UBX-MGA-GPS-EPH

Message	**UBX-MGA-GPS-EPH**					
Description	**GPS Ephemeris Assistance**					
Type	Input					
Comment	This message allows the delivery of GPS ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x00	68	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	GPS Satellite identifier (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1	-	fitInterval	-	Fit interval flag
5	U1	-	uraIndex	-	URA index
6	U1	-	svHealth	-	SV health
7	I1	2^-31	tgd	s	Group delay differential
8	U2	-	iodc	-	IODC
10	U2	2^4	toc	s	Clock data reference time
12	U1	-	reserved2	-	Reserved
13	I1	2^-55	af2	s/s squared	Time polynomial coefficient 2
14	I2	2^-43	af1	s/s	Time polynomial coefficient 1
16	I4	2^-31	af0	s	Time polynomial coefficient 0
20	I2	2^-5	crs	m	Crs
22	I2	2^-43	deltaN	semi-circles/s	Mean motion difference from computed value
24	I4	2^-31	m0	semi-circles	Mean anomaly at reference time
28	I2	2^-29	cuc	radians	Amplitude of cosine harmonic correction term to argument of latitude
30	I2	2^-29	cus	radians	Amplitude of sine harmonic correction term to argument of latitude
32	U4	2^-33	e	-	Eccentricity
36	U4	2^-19	sqrtA	m^0.5	Square root of the semi-major axis
40	U2	2^4	toe	s	Reference time of ephemeris
42	I2	2^-29	cic	radians	Amplitude of cos harmonic correction term to angle of inclination
44	I4	2^-31	omega0	semi-circles	Longitude of ascending node of orbit plane at weekly epoch

UBX-MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
48	I2	2^-29	cis	radians	Amplitude of sine harmonic correction term to angle of inclination
50	I2	2^-5	crc	m	Amplitude of cosine harmonic correction term to orbit radius
52	I4	2^-31	i0	semi-circles	Inclination angle at reference time
56	I4	2^-31	omega	semi-circles	Argument of perigee
60	I4	2^-43	omegaDot	semi-circles/s	Rate of right ascension
64	I2	2^-43	idot	semi-circles/s	Rate of inclination angle
66	U1[2]	-	reserved3	-	Reserved

25.13.8.2 UBX-MGA-GPS-ALM

Message	**UBX-MGA-GPS-ALM**					
Description	**GPS Almanac Assistance**					
Type	Input					
Comment	This message allows the delivery of GPS almanac assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x00	36	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	GPS Satellite identifier (see Satellite Numbering)
3	U1	-	svHealth	-	SV health information
4	U2	2^-21	e	-	Eccentricity
6	U1	-	almWNa	week	Reference week number of almanac (the 8 bit WNa field)
7	U1	2^12	toa	s	Reference time of almanac
8	I2	2^-19	deltaI	semi-circles	Delta inclination angle at reference time
10	I2	2^-38	omegaDot	semi-circles/s	Rate of right ascension
12	U4	2^-11	sqrtA	m^0.5	Square root of the semi-major axis

UBX-MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16	I4	2^-23	omega0	semi-circles	Longitude of ascending node of orbit plane
20	I4	2^-23	omega	semi-circles	Argument of perigee
24	I4	2^-23	m0	semi-circles	Mean anomaly at reference time
28	I2	2^-20	af0	s	Time polynomial coefficient 0 (8 MSBs)
30	I2	2^-38	af1	s/s	Time polynomial coefficient 1
32	U1[4]	-	reserved1	-	Reserved

25.13.8.3 UBX-MGA-GPS-HEALTH

Message	**UBX-MGA-GPS-HEALTH**					
Description	**GPS Health Assistance**					
Type	Input					
Comment	This message allows the delivery of GPS health assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x00	40	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x04 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	U1[32]	-	healthCode	-	Each byte represents a GPS SV (1-32). The 6 LSBs of each byte contains the 6 bit health code from subframes 4/5 page 25.
36	U1[4]	-	reserved2	-	Reserved

25.13.8.4 UBX-MGA-GPS-UTC

Message	**UBX-MGA-GPS-UTC**					
Description	**GPS UTC Assistance**					
Type	Input					
Comment	This message allows the delivery of GPS UTC assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x00	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x05 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved

UBX-MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	I4	2^-30	utcA0	s	First parameter of UTC polynomial
8	I4	2^-50	utcA1	s/s	Second parameter of UTC polynomial
12	I1	-	utcDtLS	s	Delta time due to current leap seconds
13	U1	2^12	utcTot	s	UTC parameters reference time of week (GPS time)
14	U1	-	utcWNt	weeks	UTC parameters reference week number (the 8 bit WNt field)
15	U1	-	utcWNlsf	weeks	Week number at the end of which the future leap second becomes effective (the 8 bit WNLSF field)
16	U1	-	utcDn	days	Day number at the end of which the future leap second becomes effective
17	I1	-	utcDtLSF	s	Delta time due to future leap seconds
18	U1[2]	-	reserved2	-	Reserved

25.13.8.5 UBX-MGA-GPS-IONO

Message	**UBX-MGA-GPS-IONO**				
Description	**GPS Ionosphere Assistance**				
Type	Input				
Comment	This message allows the delivery of GPS ionospheric assistance to a receiver. See the description of AssistNow Online for details.				
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x00	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x06 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I1	2^-30	ionoAlpha0	s	Ionospheric parameter alpha0 [s]
5	I1	2^-27	ionoAlpha1	s/semi-circle	Ionospheric parameter alpha1 [s/semi-circle]
6	I1	2^-24	ionoAlpha2	s/(semi-circle^2)	Ionospheric parameter alpha2 [s/semi-circle^2]
7	I1	2^-24	ionoAlpha3	s/(semi-circle^3)	Ionospheric parameter alpha3 [s/semi-circle^3]
8	I1	2^11	ionoBeta0	s	Ionospheric parameter beta0 [s]
9	I1	2^14	ionoBeta1	s/semi-circle	Ionospheric parameter beta1 [s/semi-circle]

UBX-MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
10	I1	2^16	ionoBeta2	s/(semi-circle^2)	Ionospheric parameter beta2 [s/semi-circle^2]
11	I1	2^16	ionoBeta3	s/(semi-circle^3)	Ionospheric parameter beta3 [s/semi-circle^3]
12	U1[4]	-	reserved2	-	Reserved

25.13.9 UBX-MGA-INI (0x13 0x40)

25.13.9.1 UBX-MGA-INI-POS_XYZ

Message	**UBX-MGA-INI-POS_XYZ**					
Description	**Initial Position Assistance**					
Type	Input					
Comment	**Supplying position assistance that is inaccurate by more than the specified position accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of initial position assistance to a receiver in cartesian ECEF coordinates. This message is equivalent to the UBX-MGA-INI-POS_LLH message, except for the coordinate system. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x00 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I4	-	ecefX	cm	WGS84 ECEF X coordinate
8	I4	-	ecefY	cm	WGS84 ECEF Y coordinate
12	I4	-	ecefZ	cm	WGS84 ECEF Z coordinate
16	U4	-	posAcc	cm	Position accuracy (stddev)

25.13.9.2 UBX-MGA-INI-POS_LLH

Message	**UBX-MGA-INI-POS_LLH**					
Description	**Initial Position Assistance**					
Type	Input					
Comment	**Supplying position assistance that is inaccurate by more than the specified position accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of initial position assistance to a receiver in WGS84 lat/long/alt coordinates. This message is equivalent to the UBX-MGA-INI-POS_XYZ message, except for the coordinate system. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I4	1e-7	lat	deg	WGS84 Latitude
8	I4	1e-7	lon	deg	WGS84 Longitude
12	I4	-	alt	cm	WGS84 Altitude
16	U4	-	posAcc	cm	Position accuracy (stddev)

25.13.9.3 UBX-MGA-INI-TIME_UTC

Message	**UBX-MGA-INI-TIME_UTC**					
Description	**Initial Time Assistance**					
Type	Input					
Comment	**Supplying time assistance that is inaccurate by more than the specified time accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of UTC time assistance to a receiver. This message is equivalent to the UBX-MGA-INI-TIME_GNSS message, except for the time base. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	24	see below	CK_A CK_B

Payload Contents:

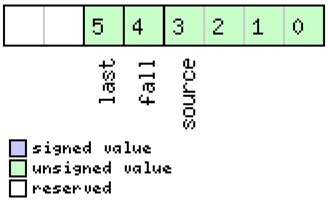
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x10 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	X1	-	ref	-	Reference to be used to set time (see graphic below)
3	I1	-	leapSecs	s	Number of leap seconds since 1980 (or 0x80 = -128 if unknown)
4	U2	-	year	-	Year
6	U1	-	month	-	Month, starting at 1

UBX-MGA-INI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
7	U1	-	day	-	Day, starting at 1
8	U1	-	hour	-	Hour, from 0 to 23
9	U1	-	minute	-	Minute, from 0 to 59
10	U1	-	second	s	Seconds, from 0 to 59
11	U1	-	reserved1	-	Reserved
12	U4	-	ns	ns	Nanoseconds, from 0 to 999,999,999
16	U2	-	tAccS	s	Seconds part of time accuracy
18	U1[2]	-	reserved2	-	Reserved
20	U4	-	tAccNs	ns	Nanoseconds part of time accuracy, from 0 to 999,999,999

Bitfield ref

This graphic explains the bits of ref



Name	Description
source	0: none, i.e. on receipt of message (will be inaccurate!) 1: relative to pulse sent to EXTINT0 2: relative to pulse sent to EXTINT1 3-15: reserved
fall	use falling edge of EXTINT pulse (default rising) - only if source is EXTINT
last	use last EXTINT pulse (default next pulse) - only if source is EXTINT

25.13.9.4 UBX-MGA-INI-TIME_GNSS

Message	**UBX-MGA-INI-TIME_GNSS**					
Description	**Initial Time Assistance**					
Type	Input					
Comment	**Supplying time assistance that is inaccurate by more than the specified time accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of time assistance to a receiver in a chosen GNSS timebase. This message is equivalent to the UBX-MGA-INI-TIME_UTC message, except for the time base. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	24	see below	CK_A CK_B

Payload Contents:

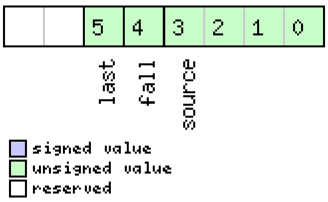
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x11 for this type)

UBX-MGA-INI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
1	U1	-	version	-	Message version (0x00 for this version)
2	X1	-	ref	-	Reference to be used to set time (see graphic below)
3	U1	-	gnssId	-	Source of time information. Currently supported: 0: GPS time 2: Galileo time 3: BeiDou time 6: GLONASS time: week = 834 + ((N4-1)*1461 + Nt)/7, tow = (((N4-1)*1461 + Nt) % 7) * 86400 + tod
4	U1[2]	-	reserved1	-	Reserved
6	U2	-	week	-	GNSS week number
8	U4	-	tow	s	GNSS time of week
12	U4	-	ns	ns	GNSS time of week, nanosecond part from 0 to 999,999,999
16	U2	-	tAccS	s	Seconds part of time accuracy
18	U1[2]	-	reserved2	-	Reserved
20	U4	-	tAccNs	ns	Nanoseconds part of time accuracy, from 0 to 999,999,999

Bitfield ref

This graphic explains the bits of ref



Name	Description
source	0: none, i.e. on receipt of message (will be inaccurate!) 1: relative to pulse sent to EXTINT0 2: relative to pulse sent to EXTINT1 3-15: reserved
fall	use falling edge of EXTINT pulse (default rising) - only if source is EXTINT
last	use last EXTINT pulse (default next pulse) - only if source is EXTINT

25.13.9.5 UBX-MGA-INI-CLKD

Message	**UBX-MGA-INI-CLKD**					
Description	**Initial Clock Drift Assistance**					
Type	Input					
Comment	**Supplying clock drift assistance that is inaccurate by more than the specified accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of clock drift assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x20 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	I4	-	clkD	ns/s	Clock drift
8	U4	-	clkDAcc	ns/s	Clock drift accuracy

25.13.9.6 UBX-MGA-INI-FREQ

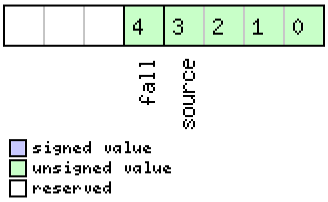
Message	**UBX-MGA-INI-FREQ**					
Description	**Initial Frequency Assistance**					
Type	Input					
Comment	**Supplying external frequency assistance that is inaccurate by more than the specified accuracy, may lead to substantially degraded receiver performance.** This message allows the delivery of external frequency assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x40	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x21 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	reserved1	-	Reserved
3	X1	-	flags	-	Frequency reference (see graphic below)
4	I4	1e-2	freq	Hz	Frequency
8	U4	-	freqAcc	ppb	Frequency accuracy

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
`source`	0: frequency available on EXTINT0 1: frequency available on EXTINT1 2-15: reserved
`fall`	use falling edge of EXTINT pulse (default rising)

25.13.9.7 UBX-MGA-INI-EOP

Message	**UBX-MGA-INI-EOP**					
Description	**Earth Orientation Parameters Assistance**					
Type	Input					
Comment	This message allows the delivery of new Earth Orientation Parameters (EOP) to a receiver to improve AssistNow Autonomous operation.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	72	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`type`	-	Message type (0x30 for this type)
1	U1	-	`version`	-	Message version (0x00 for this version)
2	U1[2]	-	`reserved1`	-	Reserved
4	U2	-	`d2kRef`	d	reference time (days since 1.1.2000 12.00h UTC)
6	U2	-	`d2kMax`	d	expiration time (days since 1.1.2000 12.00h UTC)
8	I4	2^-30	`xpP0`	arcsec	x_p t^0 polynomial term (offset)
12	I4	2^-30	`xpP1`	arcsec /d	x_p t^1 polynomial term (drift)
16	I4	2^-30	`ypP0`	arcsec	y_p t^0 polynomial term (offset)
20	I4	2^-30	`ypP1`	arcsec /d	y_p t^1 polynomial term (drift)
24	I4	2^-25	`dUT1`	s	dUT1 t^0 polynomial term (offset)
28	I4	2^-30	`ddUT1`	s/d	dUT1 t^1 polynomial term (drift)
32	U1[40]	-	`reserved2`	-	Reserved

25.13.10 UBX-MGA-QZSS (0x13 0x05)

25.13.10.1 UBX-MGA-QZSS-EPH

Message	**UBX-MGA-QZSS-EPH**				
Description	**QZSS Ephemeris Assistance**				
Type	Input				
Comment	This message allows the delivery of QZSS ephemeris assistance to a receiver. See the description of AssistNow Online for details.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x05	68	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x01 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	QZSS Satellite identifier (see Satellite Numbering), Range 1-5
3	U1	-	reserved1	-	Reserved
4	U1	-	fitInterval	-	Fit interval flag
5	U1	-	uraIndex	-	URA index
6	U1	-	svHealth	-	SV health
7	I1	2^-31	tgd	s	Group delay differential
8	U2	-	iodc	-	IODC
10	U2	2^4	toc	s	Clock data reference time
12	U1	-	reserved2	-	Reserved
13	I1	2^-55	af2	s/s squared	Time polynomial coefficient 2
14	I2	2^-43	af1	s/s	Time polynomial coefficient 1
16	I4	2^-31	af0	s	Time polynomial coefficient 0
20	I2	2^-5	crs	m	Crs
22	I2	2^-43	deltaN	semi-circles/s	Mean motion difference from computed value
24	I4	2^-31	m0	semi-circles	Mean anomaly at reference time
28	I2	2^-29	cuc	radians	Amp of cosine harmonic corr term to arg of lat
30	I2	2^-29	cus	radians	Amp of sine harmonic corr term to arg of lat
32	U4	2^-33	e	-	eccentricity
36	U4	2^-19	sqrtA	m^0.5	Square root of the semi-major axis A
40	U2	2^4	toe	s	Reference time of ephemeris
42	I2	2^-29	cic	radians	Amp of cos harmonic corr term to angle of inclination
44	I4	2^-31	omega0	semi-circles	Long of asc node of orbit plane at weekly epoch

UBX-MGA-QZSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
48	I2	2^-29	cis	radians	Amp of sine harmonic corr term to angle of inclination
50	I2	2^-5	crc	m	Amp of cosine harmonic corr term to orbit radius
52	I4	2^-31	i0	semi-circles	Inclination angle at reference time
56	I4	2^-31	omega	semi-circles	Argument of perigee
60	I4	2^-43	omegaDot	semi-circles /s	Rate of right ascension
64	I2	2^-43	idot	semi-circles /s	Rate of inclination angle
66	U1[2]	-	reserved3	-	Reserved

25.13.10.2 UBX-MGA-QZSS-ALM

Message	**UBX-MGA-QZSS-ALM**					
Description	**QZSS Almanac Assistance**					
Type	Input					
Comment	This message allows the delivery of QZSS almanac assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x05	36	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x02 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1	-	svId	-	QZSS Satellite identifier (see Satellite Numbering), Range 1-5
3	U1	-	svHealth	-	Almanac SV health information
4	U2	2^-21	e	-	Almanac eccentricity
6	U1	-	almWNa	week	Reference week number of almanac (the 8 bit WNa field)
7	U1	2^12	toa	s	Reference time of almanac
8	I2	2^-19	deltaI	semi-circles	Delta inclination angle at reference time
10	I2	2^-38	omegaDot	semi-circles /s	Almanac rate of right ascension
12	U4	2^-11	sqrtA	m^0.5	Almanac square root of the semi-major axis A

UBX-MGA-QZSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16	I4	2^-23	omega0	semi-circles	Almanac long of asc node of orbit plane at weekly
20	I4	2^-23	omega	semi-circles	Almanac argument of perigee
24	I4	2^-23	m0	semi-circles	Almanac mean anomaly at reference time
28	I2	2^-20	af0	s	Almanac time polynomial coefficient 0 (8 MSBs)
30	I2	2^-38	af1	s/s	Almanac time polynomial coefficient 1
32	U1[4]	-	reserved1	-	Reserved

25.13.10.3 UBX-MGA-QZSS-HEALTH

Message	**UBX-MGA-QZSS-HEALTH**					
Description	**QZSS Health Assistance**					
Type	Input					
Comment	This message allows the delivery of QZSS health assistance to a receiver. See the description of AssistNow Online for details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x13	0x05	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type (0x04 for this type)
1	U1	-	version	-	Message version (0x00 for this version)
2	U1[2]	-	reserved1	-	Reserved
4	U1[5]	-	healthCode	-	Each byte represents a QZSS SV (1-5). The 6 LSBs of each byte contains the 6 bit health code from subframes 4/5, data ID = 3, SV ID = 51
9	U1[3]	-	reserved2	-	Reserved

25.14 UBX-MON (0x0A)

Monitoring Messages: i.e. Communication Status, CPU Load, Stack Usage, Task Status.
Messages in the MON class are used to report the receiver status, such as CPU load, stack usage, I/O subsystem statistics etc.

25.14.1 UBX-MON-BATCH (0x0A 0x32)

25.14.1.1 Data batching buffer status

Message	**UBX-MON-BATCH**					
Description	**Data batching buffer status**					
Type	Polled					
Comment	This message contains status information about the batching buffer. It can be polled and it can also be sent by the receiver as a response to a UBX-LOG-RETRIEVEBATCH message before the UBX-LOG-BATCH messages. See Data Batching for more information.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x32	12	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (0x00 for this version)
1	U1[3]	-	`reserved1`	-	Reserved
4	U2	-	`fillLevel`	-	Current buffer fill level, i.e. number of epochs currently stored
6	U2	-	`dropsAll`	-	Number of dropped epochs since startup Note: changing the batching configuration will reset this counter.
8	U2	-	`dropsSinceMon`	-	Number of dropped epochs since last MON-BATCH message
10	U2	-	`nextMsgCnt`	-	The next retrieved UBX-LOG-BATCH will have this `msgCnt` value.

25.14.2 UBX-MON-GNSS (0x0A 0x28)

25.14.2.1 Information message major GNSS selection

Message	**UBX-MON-GNSS**					
Description	**Information message major GNSS selection**					
Type	Polled					
Comment	This message reports major GNSS selection. It does this by means of bit masks in U1 fields. Each bit in a bit mask corresponds to one major GNSS. Augmentation systems are not reported.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x28	8	see below	CK_A CK_B

Payload Contents:

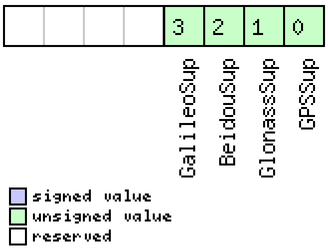
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (0x01for this version)

UBX-MON-GNSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
1	X1	-	supported	-	A bit mask showing the major GNSS that can be supported by this receiver (see graphic below)
2	X1	-	defaultGnss	-	A bit mask showing the default major GNSS selection. If the default major GNSS selection is currently configured in the efuse for this receiver, it takes precedence over the default major GNSS selection configured in the executing firmware of this receiver. (see graphic below)
3	X1	-	enabled	-	A bit mask showing the current major GNSS selection enabled for this receiver (see graphic below)
4	U1	-	simultaneous	-	Maximum number of concurrent major GNSS that can be supported by this receiver
5	U1[3]	-	reserved1	-	Reserved

Bitfield supported

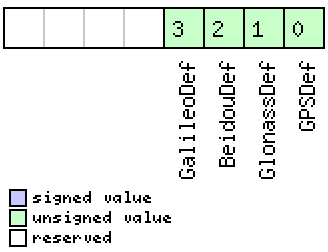
This graphic explains the bits of supported



Name	Description
GPSSup	GPS is supported
GlonassSup	GLONASS is supported
BeidouSup	BeiDou is supported
GalileoSup	Galileo is supported

Bitfield defaultGnss

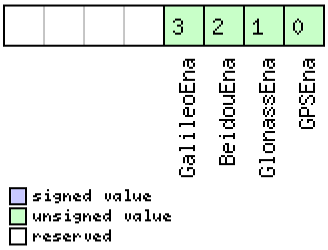
This graphic explains the bits of defaultGnss



Name	Description
GPSDef	GPS is default-enabled
GlonassDef	GLONASS is default-enabled
BeidouDef	BeiDou is default-enabled
GalileoDef	Galileo is default-enabled

Bitfield enabled

This graphic explains the bits of `enabled`



Name	Description
GPSEna	GPS is enabled
GlonassEna	GLONASS is enabled
BeidouEna	BeiDou is enabled
GalileoEna	Galileo is enabled

25.14.3 UBX-MON-HW2 (0x0A 0x0B)

25.14.3.1 Extended Hardware Status

Message	**UBX-MON-HW2**
Description	**Extended Hardware Status**
Type	Periodic/Polled
Comment	Status of different aspects of the hardware such as Imbalance, Low-Level Configuration and POST Results. The first four parameters of this message represent the complex signal from the RF front end. The following rules of thumb apply: • The smaller the absolute value of the variable `ofsI` and `ofsQ`, the better. • Ideally, the magnitude of the I-part (`magI`) and the Q-part (`magQ`) of the complex signal should be the same.

	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x0B	28	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	I1	-	ofsI	-	Imbalance of I-part of complex signal, scaled (-128 = max. negative imbalance, 127 = max. positive imbalance)
1	U1	-	magI	-	Magnitude of I-part of complex signal, scaled (0 = no signal, 255 = max. magnitude)

UBX-MON-HW2 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	I1	-	ofsQ	-	Imbalance of Q-part of complex signal, scaled (-128 = max. negative imbalance, 127 = max. positive imbalance)
3	U1	-	magQ	-	Magnitude of Q-part of complex signal, scaled (0 = no signal, 255 = max. magnitude)
4	U1	-	cfgSource	-	Source of low-level configuration (114 = ROM, 111 = OTP, 112 = config pins, 102 = flash image)
5	U1[3]	-	reserved1	-	Reserved
8	U4	-	lowLevCfg	-	Low-level configuration (obsolete in protocol versions greater than 15)
12	U1[8]	-	reserved2	-	Reserved
20	U4	-	postStatus	-	POST status word
24	U1[4]	-	reserved3	-	Reserved

25.14.4 UBX-MON-HW (0x0A 0x09)

25.14.4.1 Hardware Status

Message	**UBX-MON-HW**					
Description	**Hardware Status**					
Type	Periodic/Polled					
Comment	Status of different aspect of the hardware, such as Antenna, PIO/Peripheral Pins, Noise Level, Automatic Gain Control (AGC)					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x09	60	see below	CK_A CK_B

Payload Contents:

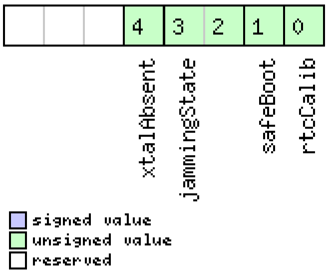
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X4	-	pinSel	-	Mask of Pins Set as Peripheral/PIO
4	X4	-	pinBank	-	Mask of Pins Set as Bank A/B
8	X4	-	pinDir	-	Mask of Pins Set as Input/Output
12	X4	-	pinVal	-	Mask of Pins Value Low/High
16	U2	-	noisePerMS	-	Noise Level as measured by the GPS Core
18	U2	-	agcCnt	-	AGC Monitor (counts SIGHI xor SIGLO, range 0 to 8191)
20	U1	-	aStatus	-	Status of the Antenna Supervisor State Machine (0=INIT, 1=DONTKNOW, 2=OK, 3=SHORT, 4=OPEN)
21	U1	-	aPower	-	Current PowerStatus of Antenna (0=OFF, 1=ON, 2=DONTKNOW)
22	X1	-	flags	-	Flags (see graphic below)
23	U1	-	reserved1	-	Reserved

UBX-MON-HW continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
24	X4	-	usedMask	-	Mask of Pins that are used by the Virtual Pin Manager
28	U1[17]	-	VP	-	Array of Pin Mappings for each of the 17 Physical Pins
45	U1	-	jamInd	-	CW Jamming indicator, scaled (0 = no CW jamming, 255 = strong CW jamming)
46	U1[2]	-	reserved2	-	Reserved
48	X4	-	pinIrq	-	Mask of Pins Value using the PIO Irq
52	X4	-	pullH	-	Mask of Pins Value using the PIO Pull High Resistor
56	X4	-	pullL	-	Mask of Pins Value using the PIO Pull Low Resistor

Bitfield flags

This graphic explains the bits of flags



Name	Description
rtcCalib	RTC is calibrated
safeBoot	safeBoot mode (0 = inactive, 1 = active)
jammingState	output from Jamming/Interference Monitor (0 = unknown or feature disabled, 1 = ok - no significant jamming, 2 = warning - interference visible but fix OK, 3 = critical - interference visible and no fix)
xtalAbsent	RTC xtal has been determined to be absent.

25.14.5 UBX-MON-IO (0x0A 0x02)

25.14.5.1 I/O Subsystem Status

Message	**UBX-MON-IO**					
Description	**I/O Subsystem Status**					
Type	Periodic/Polled					
Comment	The size of the message is determined by the number of ports 'N' the receiver supports, i.e. on u-blox 5 the number of ports is 6.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x02	0 + 20*N	see below	CK_A CK_B
Payload Contents:						

Byte Offset	Number Format	Scaling	Name	Unit	Description
Start of repeated block (N times)					

UBX-MON-IO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
N*20	U4	-	rxBytes	bytes	Number of bytes ever received
4 + 20*N	U4	-	txBytes	bytes	Number of bytes ever sent
8 + 20*N	U2	-	parityErrs	-	Number of 100ms timeslots with parity errors
10 + 20*N	U2	-	framingErrs	-	Number of 100ms timeslots with framing errors
12 + 20*N	U2	-	overrunErrs	-	Number of 100ms timeslots with overrun errors
14 + 20*N	U2	-	breakCond	-	Number of 100ms timeslots with break conditions
16 + 20*N	U1[4]	-	reserved1	-	Reserved
End of repeated block					

25.14.6 UBX-MON-MSGPP (0x0A 0x06)

25.14.6.1 Message Parse and Process Status

Message	**UBX-MON-MSGPP**					
Description	**Message Parse and Process Status**					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x06	120	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2[8]	-	msg1	msgs	Number of successfully parsed messages for each protocol on port0
16	U2[8]	-	msg2	msgs	Number of successfully parsed messages for each protocol on port1
32	U2[8]	-	msg3	msgs	Number of successfully parsed messages for each protocol on port2
48	U2[8]	-	msg4	msgs	Number of successfully parsed messages for each protocol on port3
64	U2[8]	-	msg5	msgs	Number of successfully parsed messages for each protocol on port4
80	U2[8]	-	msg6	msgs	Number of successfully parsed messages for each protocol on port5
96	U4[6]	-	skipped	bytes	Number skipped bytes for each port

25.14.7 UBX-MON-PATCH (0x0A 0x27)

25.14.7.1 Poll Request for installed patches

Message	**UBX-MON-PATCH**					
Description	**Poll Request for installed patches**					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x27	0	see below	CK_A CK_B
No payload						

25.14.7.2 Output information about installed patches.

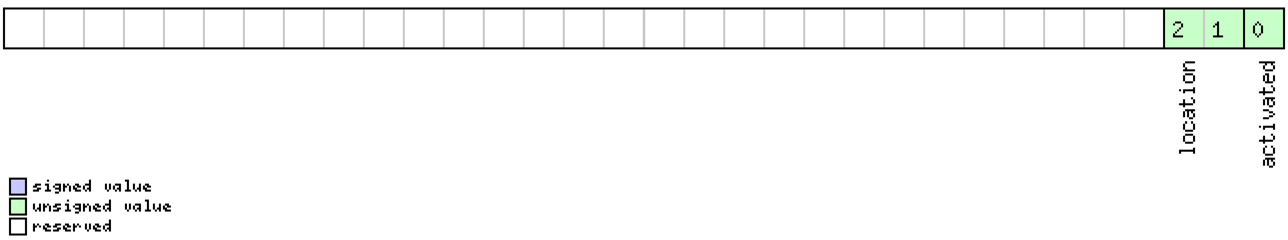
Message	**UBX-MON-PATCH**					
Description	**Output information about installed patches.**					
Type	Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x27	4 + 16*nEntries	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2	-	`version`	-	Type of the message. 0x1 for this one.
2	U2	-	`nEntries`	-	The number of patches that is output.
Start of repeated block (nEntries times)					
4 + 16*N	X4	-	`patchInfo`	-	Additional information about the patch not stated in the patch header. (see graphic below)
8 + 16*N	U4	-	`comparatorNumber`	-	The number of the comparator.
12 + 16*N	U4	-	`patchAddress`	-	The address that the targeted by the patch.
16 + 16*N	U4	-	`patchData`	-	The data that will be inserted at the patchAddress.
End of repeated block					

Bitfield patchInfo

This graphic explains the bits of `patchInfo`



Name	Description
activated	1: the patch is active. 0: otherwise.
location	Indicates where the patch is stored. 0: eFuse, 1: ROM, 2: BBR, 3: file system.

25.14.8 UBX-MON-RXBUF (0x0A 0x07)

25.14.8.1 Receiver Buffer Status

Message	**UBX-MON-RXBUF**					
Description	**Receiver Buffer Status**					
Type	Periodic/Polled					
Comment	-					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x07	24	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2[6]	-	pending	bytes	Number of bytes pending in receiver buffer for each target
12	U1[6]	-	usage	%	Maximum usage receiver buffer during the last sysmon period for each target
18	U1[6]	-	peakUsage	%	Maximum usage receiver buffer for each target

25.14.9 UBX-MON-RXR (0x0A 0x21)

25.14.9.1 Receiver Status Information

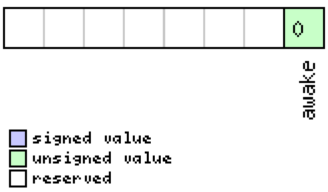
Message	**UBX-MON-RXR**					
Description	**Receiver Status Information**					
Type	Output					
Comment	The receiver ready message is sent when the receiver changes from or to backup mode.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0A	0x21	1	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X1	-	flags	-	Receiver status flags (see graphic below)

Bitfield flags

This graphic explains the bits of flags



Name	Description
awake	not in Backup mode

25.14.10 UBX-MON-TXBUF (0x0A 0x08)

25.14.10.1 Transmitter Buffer Status

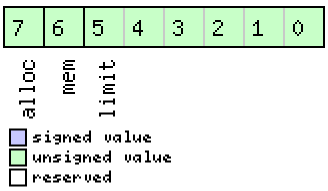
Message	UBX-MON-TXBUF					
Description	**Transmitter Buffer Status**					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x08	28	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U2[6]	-	pending	bytes	Number of bytes pending in transmitter buffer for each target
12	U1[6]	-	usage	%	Maximum usage transmitter buffer during the last sysmon period for each target
18	U1[6]	-	peakUsage	%	Maximum usage transmitter buffer for each target
24	U1	-	tUsage	%	Maximum usage of transmitter buffer during the last sysmon period for all targets
25	U1	-	tPeakusage	%	Maximum usage of transmitter buffer for all targets
26	X1	-	errors	-	Error bitmask (see graphic below)
27	U1	-	reserved1	-	Reserved

Bitfield errors

This graphic explains the bits of errors



Name	Description
limit	Buffer limit of corresponding target reached
mem	Memory Allocation error
alloc	Allocation error (TX buffer full)

25.14.11 UBX-MON-VER (0x0A 0x04)

25.14.11.1 Poll Receiver/Software Version

Message	**UBX-MON-VER**					
Description	**Poll Receiver/Software Version**					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x04	0	see below	CK_A CK_B
No payload						

25.14.11.2 Receiver/Software Version

Message	**UBX-MON-VER**					
Description	**Receiver/Software Version**					
Type	Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x04	40 + 30*N	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	CH[30]	-	`swVersion`	-	Zero-terminated Software Version String.
30	CH[10]	-	`hwVersion`	-	Zero-terminated Hardware Version String
Start of repeated block (N times)					
40 + 30*N	CH[30]	-	`extension`	-	Extended software information strings. A series of zero-terminated strings. Each extension field is 30 characters long and contains varying software information. Not all extension fields may appear. Example reported information can be: the software version string of the underlying ROM (when the receiver's firmware is running from flash), the firmware version, the supported protocol version, the module identifier, the Flash Information Structure (FIS) file information, the supported major GNSS, the supported augmentation systems.
End of repeated block					

25.15 UBX-NAV (0x01)

Navigation Results Messages: i.e. Position, Speed, Time, Acceleration, Heading, DOP, SVs used. Messages in the NAV class are used to output navigation data such as position, altitude and velocity in a number of formats. Additionally, status flags and accuracy figures are output. The messages are generated with the configured navigation/measurement rate.

25.15.1 UBX-NAV-AOPSTATUS (0x01 0x60)

25.15.1.1 AssistNow Autonomous Status

Message	**UBX-NAV-AOPSTATUS**
Description	**AssistNow Autonomous Status**
Type	Periodic/Polled
Comment	This message provides information on the status of the AssistNow Autonomous subsystem on the receiver. For example, a host application can determine the optimal time to shut down the receiver by monitoring the `status` field for a steady 0. See the chapter AssistNow Autonomous in the receiver description for details on this feature.

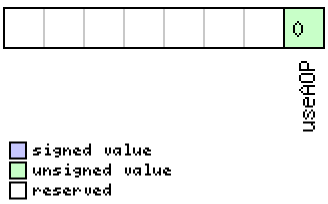
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x60	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	`aopCfg`	-	AssistNow Autonomous configuration (see graphic below)
5	U1	-	`status`	-	AssistNow Autonomous subsystem is idle (0) or running (not 0)
6	U1[10]	-	`reserved1`	-	Reserved

Bitfield aopCfg

This graphic explains the bits of `aopCfg`



Name	Description
`useAOP`	AOP enabled flag

25.15.2 UBX-NAV-CLOCK (0x01 0x22)

25.15.2.1 Clock Solution

Message	**UBX-NAV-CLOCK**					
Description	**Clock Solution**					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x22	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	clkB	ns	Clock bias
8	I4	-	clkD	ns/s	Clock drift
12	U4	-	tAcc	ns	Time accuracy estimate
16	U4	-	fAcc	ps/s	Frequency accuracy estimate

25.15.3 UBX-NAV-DGPS (0x01 0x31)

25.15.3.1 DGPS Data Used for NAV

Message	**UBX-NAV-DGPS**					
Description	**DGPS Data Used for NAV**					
Type	Periodic/Polled					
Comment	This message outputs the DGPS correction data that has been applied to the current NAV Solution. See also the notes on the RTCM protocol.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x31	16 + 12*numCh	see below	CK_A CK_B

Payload Contents:

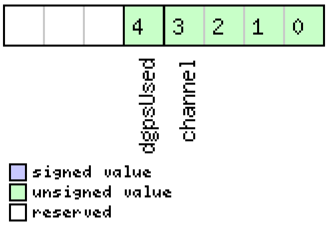
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	age	ms	Age of newest correction data
8	I2	-	baseId	-	DGPS base station identifier
10	I2	-	baseHealth	-	DGPS base station health status
12	U1	-	numCh	-	Number of channels for which correction data is following
13	U1	-	status	-	DGPS correction type status: 0x00: none 0x01: PR+PRR correction
14	U1[2]	-	reserved1	-	Reserved
Start of repeated block (numCh times)					
16 + 12*N	U1	-	svid	-	Satellite ID
17 + 12*N	X1	-	flags	-	Channel number and usage (see graphic below)

UBX-NAV-DGPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
18 + 12*N	U2	-	ageC	ms	Age of latest correction data
20 + 12*N	R4	-	prc	m	Pseudorange correction
24 + 12*N	R4	-	prrc	m/s	Pseudorange rate correction
End of repeated block					

Bitfield flags

This graphic explains the bits of flags



Name	Description
channel	GPS channel number this SV is on. Channel numbers in the firmware greater than 15 are displayed as having channel number 15
dgpsUsed	1 = DGPS used for this SV

25.15.4 UBX-NAV-DOP (0x01 0x04)

25.15.4.1 Dilution of precision

Message	**UBX-NAV-DOP**					
Description	**Dilution of precision**					
Type	Periodic/Polled					
Comment	• DOP values are dimensionless. • All DOP values are scaled by a factor of 100. If the unit transmits a value of e.g. 156, the DOP value is 1.56.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x04	18	see below	CK_A CK_B
Payload Contents:						

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U2	0.01	gDOP	-	Geometric DOP
6	U2	0.01	pDOP	-	Position DOP
8	U2	0.01	tDOP	-	Time DOP
10	U2	0.01	vDOP	-	Vertical DOP
12	U2	0.01	hDOP	-	Horizontal DOP
14	U2	0.01	nDOP	-	Northing DOP
16	U2	0.01	eDOP	-	Easting DOP

25.15.5 UBX-NAV-EOE (0x01 0x61)

25.15.5.1 End Of Epoch

Message	**UBX-NAV-EOE**					
Description	**End Of Epoch**					
Type	Periodic					
Comment	This message is intended to be used as a marker to collect all navigation messages of an epoch. It is output after all enabled NAV class messages (except UBX-NAV-HNR) and after all enabled NMEA messages.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x61	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.

25.15.6 UBX-NAV-GEOFENCE (0x01 0x39)

25.15.6.1 Geofencing status

Message	**UBX-NAV-GEOFENCE**					
Description	**Geofencing status**					
Type	Periodic/Polled					
Comment	This message outputs the evaluated states of all configured geofences for the current epoch's position. See the Geofencing description for feature details.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x39	8 + 2*numFences	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	`version`	-	Message version (0x00 for this version)
5	U1	-	`status`	-	Geofencing status 0 - Geofencing not available or not reliable 1 - Geofencing active
6	U1	-	`numFences`	-	Number of geofences
7	U1	-	`combState`	-	Combined (logical OR) state of all geofences 0 - Unknown 1 - Inside 2 - Outside
Start of repeated block (numFences times)					

UBX-NAV-GEOFENCE continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
8 + 2*N	U1	-	state	-	Geofence state 0 - Unknown 1 - Inside 2 - Outside
9 + 2*N	U1[1]	-	reserved1	-	Reserved
End of repeated block					

25.15.7 UBX-NAV-HPPOSECEF (0x01 0x13)

25.15.7.1 High Precision Position Solution in ECEF

Message	**UBX-NAV-HPPOSECEF**					
Description	**High Precision Position Solution in ECEF**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of position given in section Navigation Output Filters.** -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x13	28	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
8	I4	-	ecefX	cm	ECEF X coordinate
12	I4	-	ecefY	cm	ECEF Y coordinate
16	I4	-	ecefZ	cm	ECEF Z coordinate
20	I1	0.1	ecefXHp	mm	High precision component of ECEF X coordinate. Must be in the range of -99.. +99. Precise coordinate in cm = ecefX + (ecefXHp * 1e-2).
21	I1	0.1	ecefYHp	mm	High precision component of ECEF Y coordinate. Must be in the range of -99.. +99. Precise coordinate in cm = ecefY + (ecefYHp * 1e-2).
22	I1	0.1	ecefZHp	mm	High precision component of ECEF Z coordinate. Must be in the range of -99.. +99. Precise coordinate in cm = ecefZ + (ecefZHp * 1e-2).
23	U1	-	reserved2	-	Reserved
24	U4	0.1	pAcc	mm	Position Accuracy Estimate

25.15.8 UBX-NAV-HPPOSLLH (0x01 0x14)

25.15.8.1 High Precision Geodetic Position Solution

Message	**UBX-NAV-HPPOSLLH**				
Description	**High Precision Geodetic Position Solution**				
Type	Periodic/Polled				
Comment	**See important comments concerning validity of position given in section Navigation Output Filters.** This message outputs the Geodetic position with high precision in the currently selected ellipsoid. The default is the WGS84 Ellipsoid, but can be changed with the message UBX-CFG-DAT.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x14	36	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
8	I4	1e-7	lon	deg	Longitude
12	I4	1e-7	lat	deg	Latitude
16	I4	-	height	mm	Height above ellipsoid.
20	I4	-	hMSL	mm	Height above mean sea level
24	I1	1e-9	lonHp	deg	High precision component of longitude. Must be in the range -99..+99. Precise longitude in deg * 1e-7 = lon + (lonHp * 1e-2).
25	I1	1e-9	latHp	deg	High precision component of latitude. Must be in the range -99..+99. Precise latitude in deg * 1e-7 = lat + (latHp * 1e-2).
26	I1	0.1	heightHp	mm	High precision component of height above ellipsoid. Must be in the range -9..+9. Precise height in mm = height + (heightHp * 0.1).
27	I1	0.1	hMSLHp	mm	High precision component of height above mean sea level. Must be in range -9..+9. Precise height in mm = hMSL + (hMSLHp * 0.1)
28	U4	0.1	hAcc	mm	Horizontal accuracy estimate
32	U4	0.1	vAcc	mm	Vertical accuracy estimate

25.15.9 UBX-NAV-ODO (0x01 0x09)

25.15.9.1 Odometer Solution

Message	**UBX-NAV-ODO**					
Description	**Odometer Solution**					
Type	Periodic/Polled					
Comment	This message outputs the traveled distance since last reset (see UBX-NAV-RESETODO) together with an associated estimated accuracy and the total cumulated ground distance (can only be reset by a cold start of the receiver).					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x09	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
8	U4	-	distance	m	Ground distance since last reset
12	U4	-	totalDistance	m	Total cumulative ground distance
16	U4	-	distanceStd	m	Ground distance accuracy (1-sigma)

25.15.10 UBX-NAV-ORB (0x01 0x34)

25.15.10.1 GNSS Orbit Database Info

Message	**UBX-NAV-ORB**					
Description	**GNSS Orbit Database Info**					
Type	Periodic/Polled					
Comment	Status of the GNSS orbit database knowledge.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x34	8 + 6*numSv	see below	CK_A CK_B

Payload Contents:

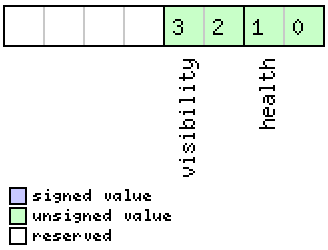
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	version	-	Message version (1, for this version)
5	U1	-	numSv	-	Number of SVs in the database
6	U1[2]	-	reserved1	-	Reserved
Start of repeated block (numSv times)					
8 + 6*N	U1	-	gnssId	-	GNSS ID
9 + 6*N	U1	-	svId	-	Satellite ID
10 + 6*N	X1	-	svFlag	-	Information Flags (see graphic below)
11 + 6*N	X1	-	eph	-	Ephemeris data (see graphic below)
12 + 6*N	X1	-	alm	-	Almanac data (see graphic below)
13 + 6*N	X1	-	otherOrb	-	Other orbit data available (see graphic below)

UBX-NAV-ORB continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
End of repeated block					

Bitfield svFlag

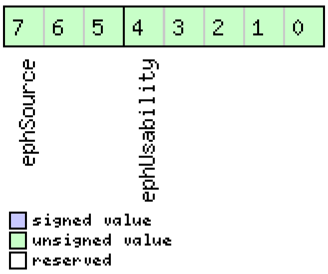
This graphic explains the bits of `svFlag`



Name	Description
`health`	SV health: 0: unknown 1: healthy 2: not healty
`visibility`	SV health: 0: unknown 1: below horizon 2: above horizon 3: above elevation mask

Bitfield eph

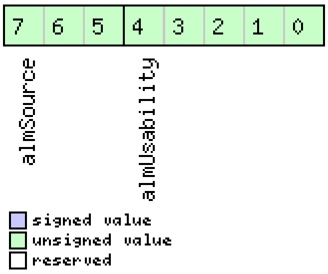
This graphic explains the bits of eph



Name	Description
`ephUsability`	How long the receiver will be able to use the stored ephemeris data from now on: 31: The usability period is unknown 30: The usability period is more than 450 minutes 30 > n > 0: The usability period is between (n-1)*15 and n*15 minutes 0: Ephemeris can no longer be used
`ephSource`	0: not available 1: GNSS transmission 2: external aiding 3-7: other

Bitfield alm

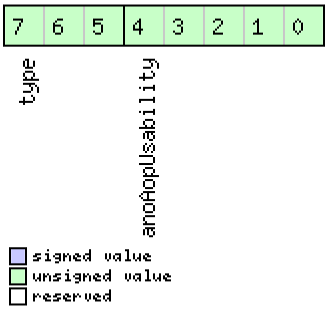
This graphic explains the bits of `alm`



Name	Description
`almUsability`	How long the receiver will be able to use the stored almanac data from now on: 31: The usability period is unknown 30: The usability period is more than 30 days 30 > n > 0: The usability period is between n-1 and n days 0: Almanac can no longer be used
`almSource`	0: not available 1: GNSS transmission 2: external aiding 3-7: other

Bitfield otherOrb

This graphic explains the bits of `otherOrb`



Name	Description
`anoAopUsability`	How long the receiver will be able to use the orbit data from now on: 31: The usability period is unknown 30: The usability period is more than 30 days 30 > n > 0: The usability period is between n-1 and n days 0: Data can no longer be used
`type`	Type of orbit data: 0: No orbit data available 1: Assist now offline data 2: Assist now autonomous data 3-7: Other orbit data

25.15.11 UBX-NAV-POSECEF (0x01 0x01)

25.15.11.1 Position Solution in ECEF

Message	**UBX-NAV-POSECEF**					
Description	**Position Solution in ECEF**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of position given in section Navigation Output Filters.** -					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x01	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	ecefX	cm	ECEF X coordinate
8	I4	-	ecefY	cm	ECEF Y coordinate
12	I4	-	ecefZ	cm	ECEF Z coordinate
16	U4	-	pAcc	cm	Position Accuracy Estimate

25.15.12 UBX-NAV-POSLLH (0x01 0x02)

25.15.12.1 Geodetic Position Solution

Message	**UBX-NAV-POSLLH**					
Description	**Geodetic Position Solution**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of position given in section Navigation Output Filters.** This message outputs the Geodetic position in the currently selected ellipsoid. The default is the WGS84 Ellipsoid, but can be changed with the message UBX-CFG-DAT.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x02	28	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	1e-7	lon	deg	Longitude
8	I4	1e-7	lat	deg	Latitude
12	I4	-	height	mm	Height above ellipsoid
16	I4	-	hMSL	mm	Height above mean sea level
20	U4	-	hAcc	mm	Horizontal accuracy estimate
24	U4	-	vAcc	mm	Vertical accuracy estimate

25.15.13 UBX-NAV-PVT (0x01 0x07)

25.15.13.1 Navigation Position Velocity Time Solution

Message	**UBX-NAV-PVT**					
Description	**Navigation Position Velocity Time Solution**					
Type	Periodic/Polled					
Comment	**Note that during a leap second there may be more (or less) than 60 seconds in a minute; see the description of leap seconds for details.** This message combines position, velocity and time solution, including accuracy figures					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x07	92	see below	CK_A CK_B

Payload Contents:

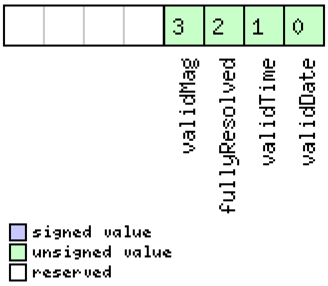
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U2	-	year	y	Year (UTC)
6	U1	-	month	month	Month, range 1..12 (UTC)
7	U1	-	day	d	Day of month, range 1..31 (UTC)
8	U1	-	hour	h	Hour of day, range 0..23 (UTC)
9	U1	-	min	min	Minute of hour, range 0..59 (UTC)
10	U1	-	sec	s	Seconds of minute, range 0..60 (UTC)
11	X1	-	valid	-	Validity flags (see graphic below)
12	U4	-	tAcc	ns	Time accuracy estimate (UTC)
16	I4	-	nano	ns	Fraction of second, range -1e9 .. 1e9 (UTC)
20	U1	-	fixType	-	GNSSfix Type: 0: no fix 1: dead reckoning only 2: 2D-fix 3: 3D-fix 4: GNSS + dead reckoning combined 5: time only fix
21	X1	-	flags	-	Fix status flags (see graphic below)
22	X1	-	flags2	-	Additional flags (see graphic below)
23	U1	-	numSV	-	Number of satellites used in Nav Solution
24	I4	1e-7	lon	deg	Longitude
28	I4	1e-7	lat	deg	Latitude
32	I4	-	height	mm	Height above ellipsoid
36	I4	-	hMSL	mm	Height above mean sea level
40	U4	-	hAcc	mm	Horizontal accuracy estimate
44	U4	-	vAcc	mm	Vertical accuracy estimate
48	I4	-	velN	mm/s	NED north velocity
52	I4	-	velE	mm/s	NED east velocity
56	I4	-	velD	mm/s	NED down velocity
60	I4	-	gSpeed	mm/s	Ground Speed (2-D)
64	I4	1e-5	headMot	deg	Heading of motion (2-D)

UBX-NAV-PVT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
68	U4	-	sAcc	mm/s	Speed accuracy estimate
72	U4	1e-5	headAcc	deg	Heading accuracy estimate (both motion and vehicle)
76	U2	0.01	pDOP	-	Position DOP
78	U1[6]	-	reserved1	-	Reserved
84	I4	1e-5	headVeh	deg	Heading of vehicle (2-D)
88	I2	1e-2	magDec	deg	Magnetic declination
90	U2	1e-2	magAcc	deg	Magnetic declination accuracy

Bitfield valid

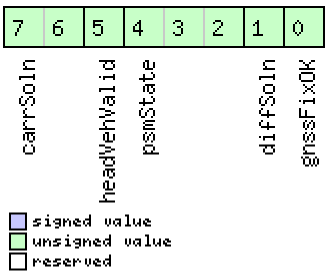
This graphic explains the bits of valid



Name	Description
validDate	1 = valid UTC Date (see Time Validity section for details)
validTime	1 = valid UTC Time of Day (see Time Validity section for details)
fullyResolved	1 = UTC Time of Day has been fully resolved (no seconds uncertainty). Cannot be used to check if time is completely solved.
validMag	1 = valid Magnetic declination

Bitfield flags

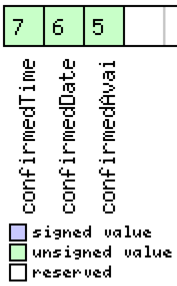
This graphic explains the bits of flags



Name	Description
gnssFixOK	1 = valid fix (i.e within DOP & accuracy masks)
diffSoln	1 = differential corrections were applied
psmState	Power Save Mode state (see Power Management): 0: PSM is not active 1: Enabled (an intermediate state before Acquisition state 2: Acquisition 3: Tracking 4: Power Optimized Tracking 5: Inactive
headVehValid	1 = heading of vehicle is valid
carrSoln	Carrier phase range solution status: 0: no carrier phase range solution 1: carrier phase range solution with floating ambiguities 2: carrier phase range solution with fixed ambiguities

Bitfield flags2

This graphic explains the bits of `flags2`



Name	Description
confirmedAvai	1 = information about UTC Date and Time of Day validity confirmation is available (see Time Validity section for details). This flag is only supported in Protocol Versions 19.00, 19.10, 20.10, 20.20, 20.30, 22.00, 23.00, 23.01, 27 and 28.
confirmedDate	1 = UTC Date validity could be confirmed (see Time Validity section for details)
confirmedTime	1 = UTC Time of Day could be confirmed (see Time Validity section for details)

25.15.14 UBX-NAV-RESETODO (0x01 0x10)

25.15.14.1 Reset odometer

Message	**UBX-NAV-RESETODO**					
Description	**Reset odometer**					
Type	Command					
Comment	This message resets the traveled distance computed by the odometer (see UBX-NAV-ODO). UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x10	0	see below	CK_A CK_B
No payload						

25.15.15 UBX-NAV-SAT (0x01 0x35)

25.15.15.1 Satellite Information

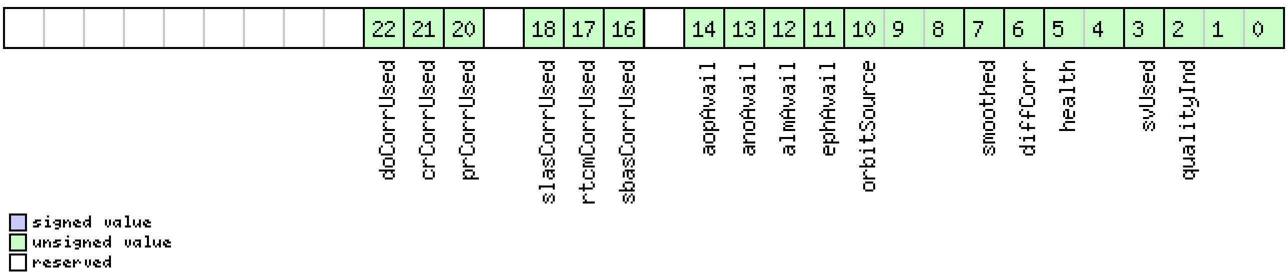
Message	**UBX-NAV-SAT**					
Description	**Satellite Information**					
Type	Periodic/Polled					
Comment	This message displays information about SVs which are either known to be visible or currently tracked by the receiver. All signal related information corresponds to the subset of signals specified in Signal Identifiers.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x35	8 + 12*numSvs	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	`version`	-	Message version (1 for this version)
5	U1	-	`numSvs`	-	Number of satellites
6	U1[2]	-	`reserved1`	-	Reserved
Start of repeated block (numSvs times)					
8 + 12*N	U1	-	`gnssId`	-	GNSS identifier (see Satellite Numbering) for assignment
9 + 12*N	U1	-	`svId`	-	Satellite identifier (see Satellite Numbering) for assignment
10 + 12*N	U1	-	`cno`	dBHz	Carrier to noise ratio (signal strength)
11 + 12*N	I1	-	`elev`	deg	Elevation (range: +/-90), unknown if out of range
12 + 12*N	I2	-	`azim`	deg	Azimuth (range 0-360), unknown if elevation is out of range
14 + 12*N	I2	0.1	`prRes`	m	Pseudorange residual
16 + 12*N	X4	-	`flags`	-	Bitmask (see graphic below)
End of repeated block					

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
qualityInd	Signal quality indicator: 0: no signal 1: searching signal 2: signal acquired 3: signal detected but unusable 4: code locked and time synchronized 5, 6, 7: code and carrier locked and time synchronized Note: Since IMES signals are not time synchronized, a channel tracking an IMES signal can never reach a quality indicator value of higher than 3.
svUsed	1 = Signal in the subset specified in Signal Identifiers is currently being used for navigation
health	Signal health flag: 0: unknown 1: healthy 2: unhealthy
diffCorr	1 = differential correction data is available for this SV
smoothed	1 = carrier smoothed pseudorange used
orbitSource	Orbit source: 0: no orbit information is available for this SV 1: ephemeris is used 2: almanac is used 3: AssistNow Offline orbit is used 4: AssistNow Autonomous orbit is used 5, 6, 7: other orbit information is used
ephAvail	1 = ephemeris is available for this SV
almAvail	1 = almanac is available for this SV
anoAvail	1 = AssistNow Offline data is available for this SV
aopAvail	1 = AssistNow Autonomous data is available for this SV
sbasCorrUsed	1 = SBAS corrections have been used for a signal in the subset specified in Signal Identifiers
rtcmCorrUsed	1 = RTCM corrections have been used for a signal in the subset specified in Signal Identifiers
slasCorrUsed	1 = QZSS SLAS corrections have been used for a signal in the subset specified in Signal Identifiers
prCorrUsed	1 = Pseudorange corrections have been used for a signal in the subset specified in Signal Identifiers
crCorrUsed	1 = Carrier range corrections have been used for a signal in the subset specified in Signal Identifiers
doCorrUsed	1 = Range rate (Doppler) corrections have been used for a signal in the subset specified in Signal Identifiers

25.15.16 UBX-NAV-SBAS (0x01 0x32)

25.15.16.1 SBAS Status Data

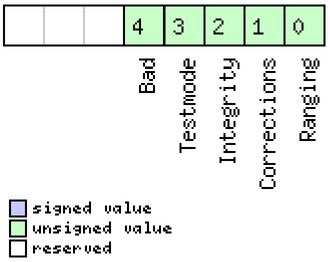
Message	**UBX-NAV-SBAS**					
Description	**SBAS Status Data**					
Type	Periodic/Polled					
Comment	This message outputs the status of the SBAS sub system					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x32	12 + 12*cnt	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	geo	-	PRN Number of the GEO where correction and integrity data is used from
5	U1	-	mode	-	SBAS Mode 0 Disabled 1 Enabled Integrity 3 Enabled Testmode
6	I1	-	sys	-	SBAS System (WAAS/EGNOS/...) -1 Unknown 0 WAAS 1 EGNOS 2 MSAS 3 GAGAN 16 GPS
7	X1	-	service	-	SBAS Services available (see graphic below)
8	U1	-	cnt	-	Number of SV data following
9	U1[3]	-	reserved1	-	Reserved
Start of repeated block (cnt times)					
12 + 12*N	U1	-	svid	-	SV ID
13 + 12*N	U1	-	flags	-	Flags for this SV
14 + 12*N	U1	-	udre	-	Monitoring status
15 + 12*N	U1	-	svSys	-	System (WAAS/EGNOS/...) same as SYS
16 + 12*N	U1	-	svService	-	Services available same as SERVICE
17 + 12*N	U1	-	reserved2	-	Reserved
18 + 12*N	I2	-	prc	cm	Pseudo Range correction in [cm]
20 + 12*N	U1[2]	-	reserved3	-	Reserved
22 + 12*N	I2	-	ic	cm	Ionosphere correction in [cm]
End of repeated block					

Bitfield service

This graphic explains the bits of service



Name	Description
Ranging	GEO may be used as ranging source
Corrections	GEO is providing correction data
Integrity	GEO is providing integrity
Testmode	GEO is in test mode
Bad	Problem with signal or broadcast data indicated

25.15.17 UBX-NAV-SOL (0x01 0x06)

25.15.17.1 Navigation Solution Information

Message	**UBX-NAV-SOL**				
Description	**Navigation Solution Information**				
Type	Periodic/Polled				
Comment	This message combines position, velocity and time solution in ECEF, including accuracy figures. This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-PVT message in preference.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x06	52	see below	CK_A CK_B

Payload Contents:

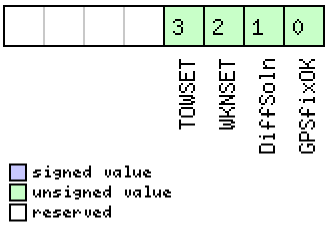
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	fTOW	ns	Fractional part of iTOW (range: +/-500000). The precise GPS time of week in seconds is: (iTOW * 1e-3) + (fTOW * 1e-9)
8	I2	-	week	weeks	GPS week number of the navigation epoch
10	U1	-	gpsFix	-	GPSfix Type, range 0..5 0x00 = No Fix 0x01 = Dead Reckoning only 0x02 = 2D-Fix 0x03 = 3D-Fix 0x04 = GPS + dead reckoning combined 0x05 = Time only fix 0x06..0xff: reserved
11	X1	-	flags	-	Fix Status Flags (see graphic below)

UBX-NAV-SOL continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
12	I4	-	ecefX	cm	ECEF X coordinate
16	I4	-	ecefY	cm	ECEF Y coordinate
20	I4	-	ecefZ	cm	ECEF Z coordinate
24	U4	-	pAcc	cm	3D Position Accuracy Estimate
28	I4	-	ecefVX	cm/s	ECEF X velocity
32	I4	-	ecefVY	cm/s	ECEF Y velocity
36	I4	-	ecefVZ	cm/s	ECEF Z velocity
40	U4	-	sAcc	cm/s	Speed Accuracy Estimate
44	U2	0.01	pDOP	-	Position DOP
46	U1	-	reserved1	-	Reserved
47	U1	-	numSV	-	Number of SVs used in Nav Solution
48	U1[4]	-	reserved2	-	Reserved

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
GPSfixOK	1 = Fix within limits (e.g. DOP & accuracy)
DiffSoln	1 = DGPS used
WKNSET	1 = Valid GPS week number (see Time Validity section for details)
TOWSET	1 = Valid GPS time of week (iTOW & fTOW, see Time Validity section for details)

25.15.18 UBX-NAV-STATUS (0x01 0x03)

25.15.18.1 Receiver Navigation Status

Message	**UBX-NAV-STATUS**					
Description	**Receiver Navigation Status**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of position and velocity given in section Navigation Output Filters.** -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x03	16	see below	CK_A CK_B

Payload Contents:

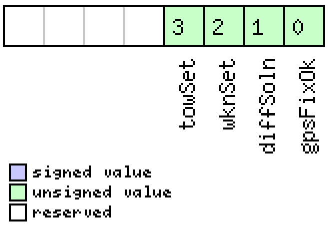
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.

UBX-NAV-STATUS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	U1	-	gpsFix	-	GPSfix Type, this value does **not** qualify a fix as valid and within the limits. See note on flag gpsFixOk below. 0x00 = no fix 0x01 = dead reckoning only 0x02 = 2D-fix 0x03 = 3D-fix 0x04 = GPS + dead reckoning combined 0x05 = Time only fix 0x06..0xff = reserved
5	X1	-	flags	-	Navigation Status Flags (see graphic below)
6	X1	-	fixStat	-	Fix Status Information (see graphic below)
7	X1	-	flags2	-	further information about navigation output (see graphic below)
8	U4	-	ttff	ms	Time to first fix (millisecond time tag)
12	U4	-	msss	ms	Milliseconds since Startup / Reset

Bitfield flags

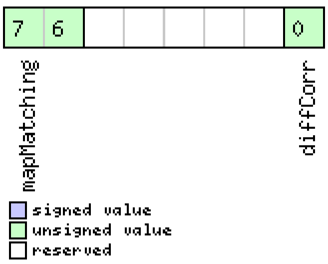
This graphic explains the bits of flags



Name	Description
gpsFixOk	1 = position and velocity valid and within DOP and ACC Masks, see also important comments in section Navigation Output Filters.
diffSoln	1 = differential corrections were applied
wknSet	1 = Week Number valid (see Time Validity section for details)
towSet	1 = Time of Week valid (see Time Validity section for details)

Bitfield fixStat

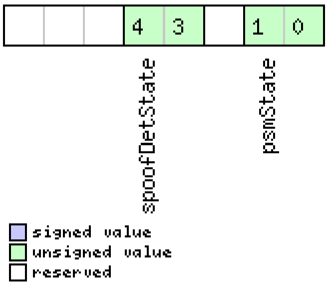
This graphic explains the bits of fixStat



Name	Description
diffCorr	1 = differential corrections available
mapMatching	map matching status: 00: none 01: valid but not used, i.e. map matching data was received, but was too old 10: valid and used, map matching data was applied 11: valid and used, map matching data was applied. In case of sensor unavailability map matching data enables dead reckoning. This requires map matched latitude/longitude or heading data.

Bitfield flags2

This graphic explains the bits of `flags2`



Name	Description
psmState	power save mode state 0: ACQUISITION [or when psm disabled] 1: TRACKING 2: POWER OPTIMIZED TRACKING 3: INACTIVE
spoofDetState	Spoofing detection state 0: Unknown or deactivated 1: No spoofing indicated 2: Spoofing indicated 3: Multiple spoofing indications Note that the spoofing state value only reflects the dector state for the current navigation epoch. As spoofing can be detected most easily at the transition from real signal to spoofing signal, this is also where the detector is triggered the most. I.e. a value of 1 - No spoofing indicated does not mean that the receiver is not spoofed, it simply states that the detector was not triggered in this epoch.

25.15.19 UBX-NAV-SVINFO (0x01 0x30)

25.15.19.1 Space Vehicle Information

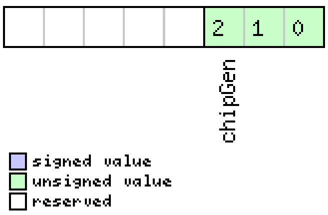
Message	**UBX-NAV-SVINFO**					
Description	**Space Vehicle Information**					
Type	Periodic/Polled					
Comment	Information about satellites used or visible This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-SAT message in preference.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x30	8 + 12*numCh	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	numCh	-	Number of channels
5	X1	-	globalFlags	-	Bitmask (see graphic below)
6	U1[2]	-	reserved1	-	Reserved
Start of repeated block (numCh times)					
8 + 12*N	U1	-	chn	-	Channel number, 255 for SVs not assigned to a channel
9 + 12*N	U1	-	svid	-	Satellite ID, see Satellite Numbering for assignment
10 + 12*N	X1	-	flags	-	Bitmask (see graphic below)
11 + 12*N	X1	-	quality	-	Bitfield (see graphic below)
12 + 12*N	U1	-	cno	dBHz	Carrier to Noise Ratio (Signal Strength)
13 + 12*N	I1	-	elev	deg	Elevation in integer degrees
14 + 12*N	I2	-	azim	deg	Azimuth in integer degrees
16 + 12*N	I4	-	prRes	cm	Pseudo range residual in centimeters
End of repeated block					

Bitfield globalFlags

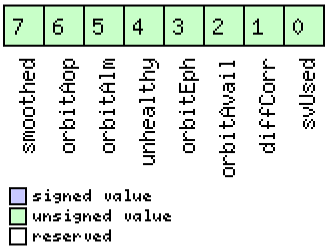
This graphic explains the bits of `globalFlags`



Name	Description
chipGen	Chip hardware generation 0: Antaris, Antaris 4 1: u-blox 5 2: u-blox 6 3: u-blox 7 4: u-blox 8 / u-blox M8

Bitfield flags

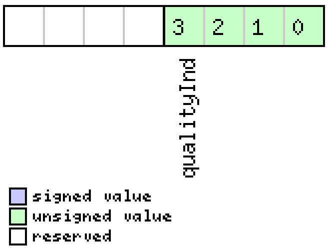
This graphic explains the bits of `flags`



Name	Description
svUsed	SV is used for navigation
diffCorr	Differential correction data is available for this SV
orbitAvail	Orbit information is available for this SV (Ephemeris or Almanac)
orbitEph	Orbit information is Ephemeris
unhealthy	SV is unhealthy / shall not be used
orbitAlm	Orbit information is Almanac Plus
orbitAop	Orbit information is AssistNow Autonomous
smoothed	Carrier smoothed pseudorange used

Bitfield quality

This graphic explains the bits of `quality`



Name	Description
qualityInd	Signal Quality indicator (range 0..7). The following list shows the meaning of the different QI values: 0: no signal 1: searching signal 2: signal acquired 3: signal detected but unusable 4: code locked and time synchronized 5, 6, 7: code and carrier locked and time synchronized Note: Since IMES signals are not time synchronized, a channel tracking an IMES signal can never reach a quality indicator value of higher than 3.

25.15.20 UBX-NAV-TIMEBDS (0x01 0x24)

25.15.20.1 BDS Time Solution

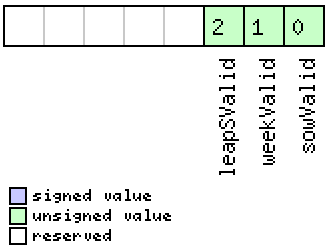
Message	**UBX-NAV-TIMEBDS**					
Description	**BDS Time Solution**					
Type	Periodic/Polled					
Comment	This message reports the precise BDS time of the most recent navigation solution including validity flags and an accuracy estimate.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x24	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U4	-	SOW	s	BDS time of week (rounded to seconds)
8	I4	-	fSOW	ns	Fractional part of SOW (range: +/-500000000). The precise BDS time of week in seconds is: SOW + fSOW * 1e-9
12	I2	-	week	-	BDS week number of the navigation epoch
14	I1	-	leapS	s	BDS leap seconds (BDS-UTC)
15	X1	-	valid	-	Validity Flags (see graphic below)
16	U4	-	tAcc	ns	Time Accuracy Estimate

Bitfield valid

This graphic explains the bits of valid



Name	Description
sowValid	1 = Valid SOW and fSOW (see Time Validity section for details)
weekValid	1 = Valid week (see Time Validity section for details)
leapSValid	1 = Valid leapS

25.15.21 UBX-NAV-TIMEGAL (0x01 0x25)

25.15.21.1 Galileo Time Solution

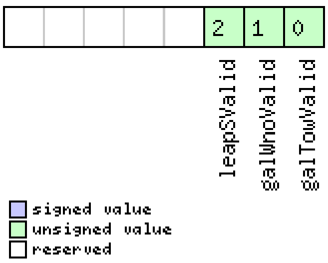
Message	**UBX-NAV-TIMEGAL**					
Description	**Galileo Time Solution**					
Type	Periodic/Polled					
Comment	This message reports the precise Galileo time of the most recent navigation solution including validity flags and an accuracy estimate.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x25	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U4	-	`galTow`	s	Galileo time of week (rounded to seconds)
8	I4	-	`fGalTow`	ns	Fractional part of the Galileo time of week (range: +/-500000000). The precise Galileo time of week in seconds is: `galTow + fGalTow * 1e-9`
12	I2	-	`galWno`	-	Galileo week number
14	I1	-	`leapS`	s	Galileo leap seconds (Galileo-UTC)
15	X1	-	`valid`	-	Validity Flags (see graphic below)
16	U4	-	`tAcc`	ns	Time Accuracy Estimate

Bitfield valid

This graphic explains the bits of `valid`



Name	Description
`galTowValid`	1 = Valid galTow and fGalTow (see Time Validity section for details)
`galWnoValid`	1 = Valid galWno (see Time Validity section for details)
`leapSValid`	1 = Valid leapS

25.15.22 UBX-NAV-TIMEGLO (0x01 0x23)

25.15.22.1 GLO Time Solution

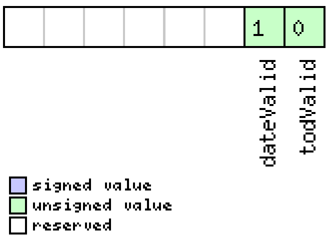
Message	**UBX-NAV-TIMEGLO**					
Description	**GLO Time Solution**					
Type	Periodic/Polled					
Comment	This message reports the precise GLO time of the most recent navigation solution including validity flags and an accuracy estimate.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x23	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	`iTOW`	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U4	-	`TOD`	s	GLONASS time of day (rounded to integer seconds)
8	I4	-	`fTOD`	ns	Fractional part of TOD (range: +/-500000000). The precise GLONASS time of day in seconds is: `TOD + fTOD * 1e-9`
12	U2	-	`Nt`	days	Current date (range: 1-1461), starting at 1 from the 1st Jan of the year indicated by N4 and ending at 1461 at the 31st Dec of the third year after that indicated by N4
14	U1	-	`N4`	-	Four-year interval number starting from 1996 (1=1996, 2=2000, 3=2004...)
15	X1	-	`valid`	-	Validity flags (see graphic below)
16	U4	-	`tAcc`	ns	Time Accuracy Estimate

Bitfield valid

This graphic explains the bits of `valid`



Name	Description
todValid	1 = Valid TOD and fTOD (see Time Validity section for details)
dateValid	1 = Valid N4 and Nt (see Time Validity section for details)

25.15.23 UBX-NAV-TIMEGPS (0x01 0x20)

25.15.23.1 GPS Time Solution

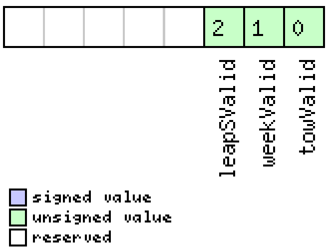
Message	**UBX-NAV-TIMEGPS**					
Description	**GPS Time Solution**					
Type	Periodic/Polled					
Comment	This message reports the precise GPS time of the most recent navigation solution including validity flags and an accuracy estimate.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x20	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	fTOW	ns	Fractional part of iTOW (range: +/-500000). The precise GPS time of week in seconds is: (iTOW * 1e-3) + (fTOW * 1e-9)
8	I2	-	week	-	GPS week number of the navigation epoch
10	I1	-	leapS	s	GPS leap seconds (GPS-UTC)
11	X1	-	valid	-	Validity Flags (see graphic below)
12	U4	-	tAcc	ns	Time Accuracy Estimate

Bitfield valid

This graphic explains the bits of valid



Name	Description
towValid	1 = Valid GPS time of week (iTOW & fTOW, see Time Validity section for details)
weekValid	1 = Valid GPS week number (see Time Validity section for details)
leapSValid	1 = Valid GPS leap seconds

25.15.24 UBX-NAV-TIMELS (0x01 0x26)

25.15.24.1 Leap second event information

Message	**UBX-NAV-TIMELS**					
Description	**Leap second event information**					
Type	Periodic/Polled					
Comment	Information about the upcoming leap second event if one is scheduled.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x26	24	see below	CK_A CK_B

Payload Contents:

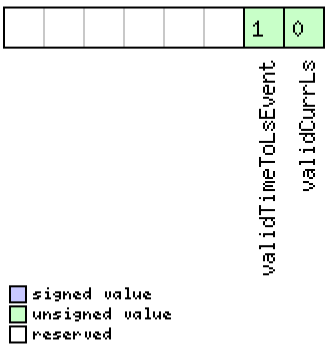
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U1	-	version	-	Message version (0x00 for this version).
5	U1[3]	-	reserved1	-	Reserved
8	U1	-	srcOfCurrLs	-	Information source for the current number of leap seconds. 0: Default (hardcoded in the firmware, can be outdated) 1: Derived from time difference between GPS and GLONASS time 2: GPS 3: SBAS 4: BeiDou 5: Galileo 6: Aided data 7: Configured 255: Unknown
9	I1	-	currLs	s	Current number of leap seconds since start of GPS time (Jan 6, 1980). It reflects how much GPS time is ahead of UTC time. Galileo number of leap seconds is the same as GPS. BeiDou number of leap seconds is 14 less than GPS. GLONASS follows UTC time, so no leap seconds.

UBX-NAV-TIMELS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
10	U1	-	srcOfLsChange	-	Information source for the future leap second event. 0: No source 2: GPS 3: SBAS 4: BeiDou 5: Galileo 6: GLONASS
11	I1	-	lsChange	s	Future leap second change if one is scheduled. +1 = positive leap second, -1 = negative leap second, 0 = no future leap second event scheduled or no information available.
12	I4	-	timeToLsEvent	s	Number of seconds until the next leap second event, or from the last leap second event if no future event scheduled. If > 0 event is in the future, = 0 event is now, < 0 event is in the past. Valid only if validTimeToLsEvent = 1.
16	U2	-	dateOfLsGpsWn	-	GPS week number (WN) of the next leap second event or the last one if no future event scheduled. Valid only if validTimeToLsEvent = 1.
18	U2	-	dateOfLsGpsDn	-	GPS day of week number (DN) for the next leap second event or the last one if no future event scheduled. Valid only if validTimeToLsEvent = 1. (GPS and Galileo DN: from 1 = Sun to 7 = Sat. BeiDou DN: from 0 = Sun to 6 = Sat.)
20	U1[3]	-	reserved2	-	Reserved
23	X1	-	valid	-	Validity flags (see graphic below)

Bitfield valid

This graphic explains the bits of valid



Name	Description
validCurrLs	1 = Valid current number of leap seconds value.
validTimeToLsEvent	1 = Valid time to next leap second event or from the last leap second event if no future event scheduled.

25.15.25 UBX-NAV-TIMEUTC (0x01 0x21)

25.15.25.1 UTC Time Solution

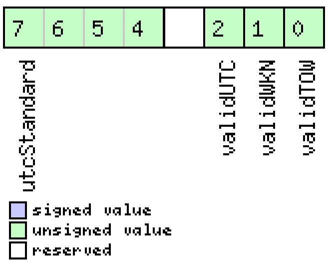
Message	**UBX-NAV-TIMEUTC**					
Description	**UTC Time Solution**					
Type	Periodic/Polled					
Comment	**Note that during a leap second there may be more or less than 60 seconds in a minute; see the description of leap seconds for details.** -					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x21	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	U4	-	tAcc	ns	Time accuracy estimate (UTC)
8	I4	-	nano	ns	Fraction of second, range -1e9 .. 1e9 (UTC)
12	U2	-	year	y	Year, range 1999..2099 (UTC)
14	U1	-	month	month	Month, range 1..12 (UTC)
15	U1	-	day	d	Day of month, range 1..31 (UTC)
16	U1	-	hour	h	Hour of day, range 0..23 (UTC)
17	U1	-	min	min	Minute of hour, range 0..59 (UTC)
18	U1	-	sec	s	Seconds of minute, range 0..60 (UTC)
19	X1	-	valid	-	Validity Flags (see graphic below)

Bitfield valid

This graphic explains the bits of valid



Name	Description
validTOW	1 = Valid Time of Week (see Time Validity section for details)
validWKN	1 = Valid Week Number (see Time Validity section for details)
validUTC	1 = Valid UTC Time
utcStandard	UTC standard identifier. 0: Information not available 1: Communications Research Labratory (CRL) 2: National Institute of Standards and Technology (NIST) 3: U.S. Naval Observatory (USNO) 4: International Bureau of Weights and Measures (BIPM) 5: European Laboratory (tbd) 6: Former Soviet Union (SU) 7: National Time Service Center, China (NTSC) 15: Unknown

25.15.26 UBX-NAV-VELECEF (0x01 0x11)

25.15.26.1 Velocity Solution in ECEF

Message	**UBX-NAV-VELECEF**					
Description	**Velocity Solution in ECEF**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of velocity given in section Navigation Output Filters.** -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x11	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	ecefVX	cm/s	ECEF X velocity
8	I4	-	ecefVY	cm/s	ECEF Y velocity
12	I4	-	ecefVZ	cm/s	ECEF Z velocity
16	U4	-	sAcc	cm/s	Speed accuracy estimate

25.15.27 UBX-NAV-VELNED (0x01 0x12)

25.15.27.1 Velocity Solution in NED

Message	**UBX-NAV-VELNED**					
Description	**Velocity Solution in NED**					
Type	Periodic/Polled					
Comment	**See important comments concerning validity of velocity given in section Navigation Output Filters.** -					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x01	0x12	36	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I4	-	velN	cm/s	North velocity component
8	I4	-	velE	cm/s	East velocity component
12	I4	-	velD	cm/s	Down velocity component
16	U4	-	speed	cm/s	Speed (3-D)
20	U4	-	gSpeed	cm/s	Ground speed (2-D)
24	I4	1e-5	heading	deg	Heading of motion 2-D
28	U4	-	sAcc	cm/s	Speed accuracy Estimate
32	U4	1e-5	cAcc	deg	Course / Heading accuracy estimate

25.16 UBX-RXM (0x02)

Receiver Manager Messages: i.e. Satellite Status, RTC Status.
Messages in the RXM class are used to output status and result data from the Receiver Manager.

25.16.1 UBX-RXM-IMES (0x02 0x61)

25.16.1.1 Indoor Messaging System Information

Message	**UBX-RXM-IMES**					
Description	**Indoor Messaging System Information**					
Type	Periodic/Polled					
Comment	This message shows the IMES stations the receiver is currently tracking, their data rate, the signal level, the Doppler (with respect to 1575.4282MHz) and what data (without protocol specific overhead) it has received from these stations so far. This message is sent out at the navigation rate the receiver is currently set to. Therefore it allows users to get an overview on the receiver's current state from the IMES perspective.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x61	4 + 44*numTx	see below	CK_A CK_B

Payload Contents:

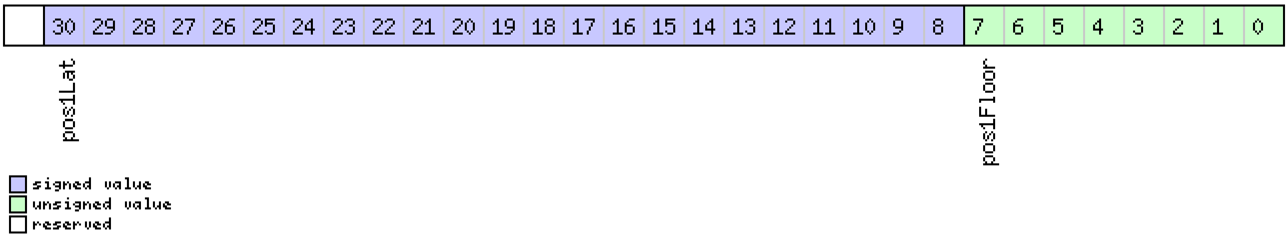
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	numTx	-	Number of transmitters contained in the message
1	U1	-	version	-	Message version (0x01 for this version)
2	U1[2]	-	reserved1	-	Reserved
Start of repeated block (numTx times)					
4 + 44*N	U1	-	reserved2	-	Reserved
5 + 44*N	U1	-	txId	-	Transmitter identifier
6 + 44*N	U1[3]	-	reserved3	-	Reserved
9 + 44*N	U1	-	cno	dBHz	Carrier to Noise Ratio (Signal Strength)
10 + 44*N	U1[2]	-	reserved4	-	Reserved
12 + 44*N	I4	2^-12	doppler	Hz	Doppler frequency with respect to 1575.4282MHz [IIIII.FFF Hz]
16 + 44*N	X4	-	position1_1	-	Position 1 Frame (part 1/2) (see graphic below)
20 + 44*N	X4	-	position1_2	-	Position 1 Frame (part 2/2) (see graphic below)
24 + 44*N	X4	-	position2_1	-	Position 2 Frame (part 1/3) (see graphic below)
28 + 44*N	I4	180*2^-24	lat	deg	Latitude, Position 2 Frame (part 2/3)
32 + 44*N	I4	360*2^-25	lon	deg	Longitude, Position 2 Frame (part 3/3)
36 + 44*N	X4	-	shortIdFrame	-	Short ID Frame (see graphic below)
40 + 44*N	U4	-	mediumIdLSB	-	Medium ID LSB, Medium ID Frame (part 1/2)

UBX-RXM-IMES continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
44 + 44*N	X4	-	mediumId_2	-	Medium ID Frame (part 2/2) (see graphic below)
End of repeated block					

Bitfield position1_1

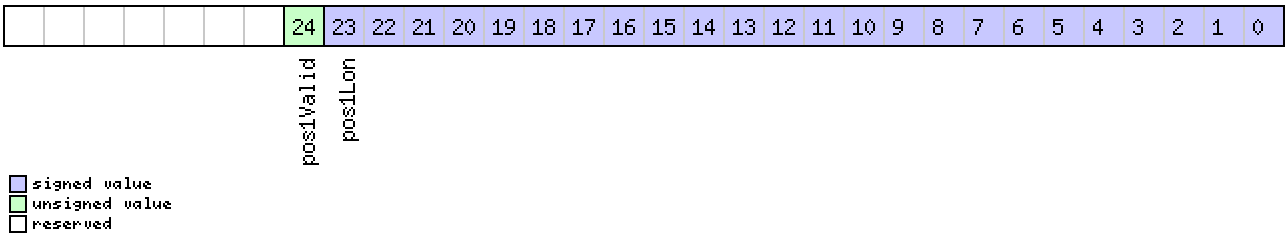
This graphic explains the bits of position1_1



Name	Description
pos1Floor	Floor number [1.0 floor resolution] (Offset: -50 floor)
pos1Lat	Latitude [deg * (180 / 2^23)]

Bitfield position1_2

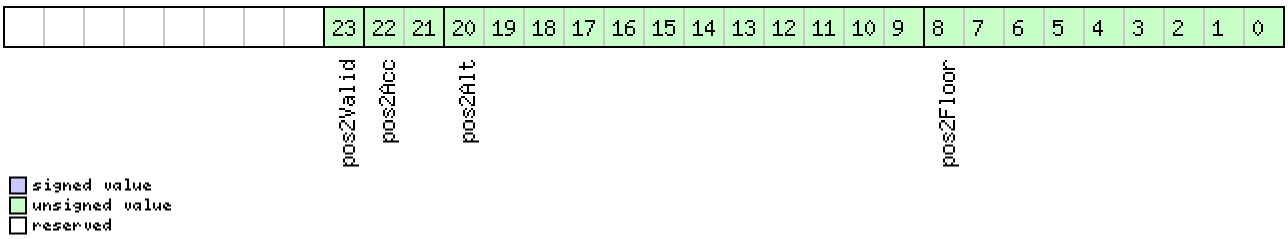
This graphic explains the bits of position1_2



Name	Description
pos1Lon	Longitude [deg * (360 / 2^24)]
pos1Valid	Position 1 Frame valid

Bitfield position2_1

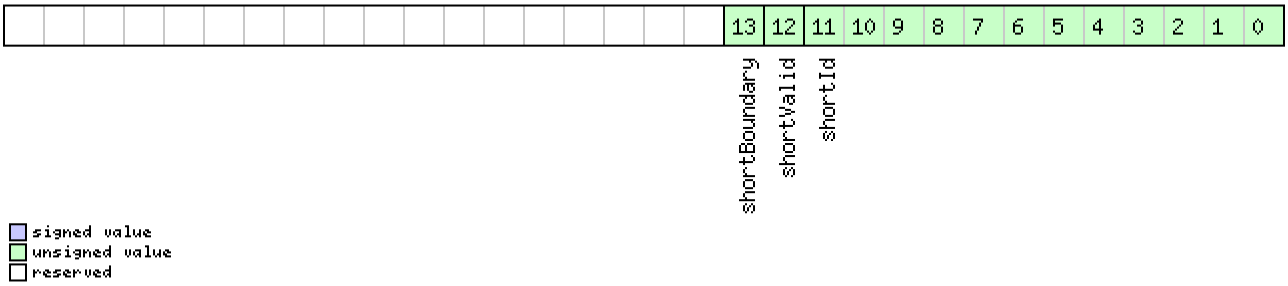
This graphic explains the bits of position2_1



Name	Description
pos2Floor	Floor number [0.5 floor resolution] (Offset: -50 floor)
pos2Alt	Altitude [m] (Offset: -95m)
pos2Acc	Accuracy Index (0:undef, 1:<7m, 2:<15m, 3:>15m)
pos2Valid	Position 2 Frame valid

Bitfield shortIdFrame

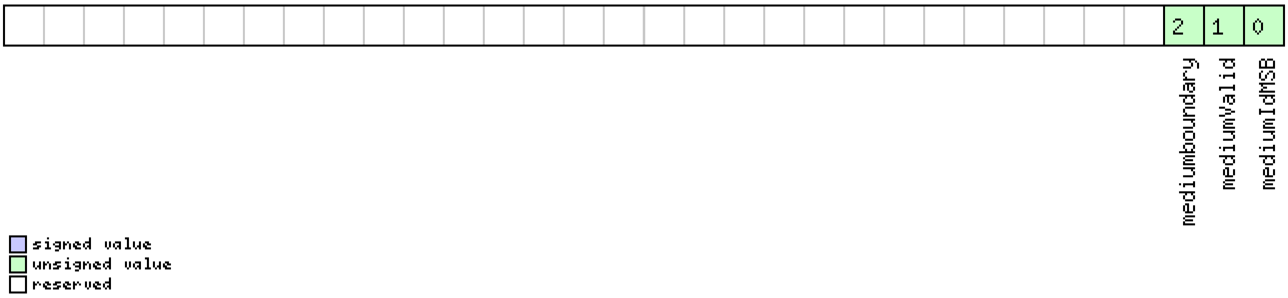
This graphic explains the bits of shortIdFrame



Name	Description
shortId	Short ID
shortValid	Short ID Frame valid
shortBoundary	Boundary Bit

Bitfield mediumId_2

This graphic explains the bits of mediumId_2



Name	Description
mediumIdMSB	Medium ID MSB
mediumValid	Medium ID Frame valid
mediumboundary	Boundary Bit

25.16.2 UBX-RXM-MEASX (0x02 0x14)

25.16.2.1 Satellite Measurements for RRLP

Message	**UBX-RXM-MEASX**					
Description	**Satellite Measurements for RRLP**					
Type	Periodic					
Comment	The message payload data is, where possible and appropriate, according to the Radio Resource LCS (Location Services) Protocol (RRLP) [1]. One exception is the satellite and GNSS ids, which here are given according to the Satellite Numbering scheme. The correct satellites have to be selected and their satellite ID translated accordingly [1, tab. A.10.14] for use in a RRLP Measure Position Response Component. Similarly, the measurement reference time of week has to be forwarded correctly (modulo 14400000 for the 24 LSB GPS measurements variant, modulo 3600000 for the 22 LSB Galileo and Additional Navigation Satelllite Systems (GANSS) measurements variant) of the RRLP measure position response to the SMLC. Reference: [1] ETSI TS 144 031 V11.0.0 (2012-10), Digital cellular telecommunications system (Phase 2+), Location Services (LCS), Mobile Station (MS) - Serving Mobile Location Centre (SMLC), Radio Resource LCS Protocol (RRLP), (3GPP TS 44.031 version 11.0.0 Release 11).					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x14	44 + 24*numSV	see below	CK_A CK_B

Payload Contents:

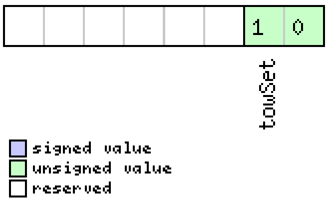
Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version, currently 0x01
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	gpsTOW	ms	GPS measurement reference time
8	U4	-	gloTOW	ms	GLONASS measurement reference time
12	U4	-	bdsTOW	ms	BeiDou measurement reference time
16	U1[4]	-	reserved2	-	Reserved
20	U4	-	qzssTOW	ms	QZSS measurement reference time
24	U2	2^-4	gpsTOWacc	ms	GPS measurement reference time accuracy (0xffff = > 4s)
26	U2	2^-4	gloTOWacc	ms	GLONASS measurement reference time accuracy (0xffff = > 4s)
28	U2	2^-4	bdsTOWacc	ms	BeiDou measurement reference time accuracy (0xffff = > 4s)
30	U1[2]	-	reserved3	-	Reserved
32	U2	2^-4	qzssTOWacc	ms	QZSS measurement reference time accuracy (0xffff = > 4s)
34	U1	-	numSV	-	Number of satellites in repeated block
35	U1	-	flags	-	Flags (see graphic below)
36	U1[8]	-	reserved4	-	Reserved
Start of repeated block (numSV times)					
44 + 24*N	U1	-	gnssId	-	GNSS ID (see Satellite Numbering)
45 + 24*N	U1	-	svId	-	Satellite ID (see Satellite Numbering)

UBX-RXM-MEASX continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
46 + 24*N	U1	-	cNo	-	carrier noise ratio (0..63)
47 + 24*N	U1	-	mpathIndic	-	multipath index (according to [1]) (0 = not measured, 1 = low, 2 = medium, 3 = high)
48 + 24*N	I4	0.04	dopplerMS	m/s	Doppler measurement
52 + 24*N	I4	0.2	dopplerHz	Hz	Doppler measurement
56 + 24*N	U2	-	wholeChips	-	whole value of the code phase measurement (0..1022 for GPS)
58 + 24*N	U2	-	fracChips	-	fractional value of the code phase measurement (0..1023)
60 + 24*N	U4	2^-21	codePhase	ms	Code phase
64 + 24*N	U1	-	intCodePhase	ms	Integer (part of the) code phase
65 + 24*N	U1	-	pseuRangeRMSErr	-	pseudorange RMS error index (according to [1]) (0..63)
66 + 24*N	U1[2]	-	reserved5	-	Reserved
End of repeated block					

Bitfield flags

This graphic explains the bits of flags



Name	Description
towSet	TOW set (0 = no, 1 or 2 = yes)

25.16.3 UBX-RXM-PMREQ (0x02 0x41)

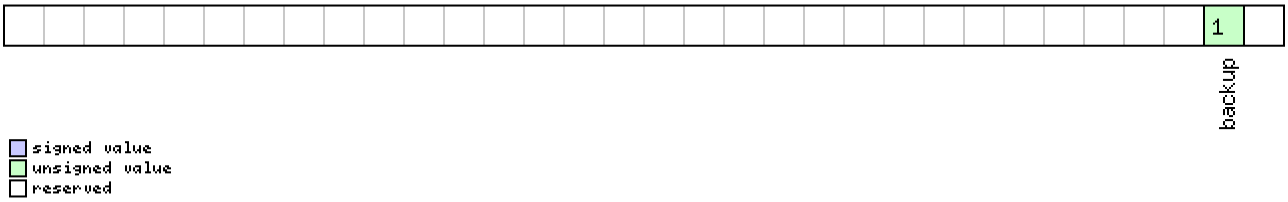
25.16.3.1 Requests a Power Management task

Message	**UBX-RXM-PMREQ**					
Description	**Requests a Power Management task**					
Type	Command					
Comment	Request of a Power Management related task of the receiver.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x41	8	see below	CK_A CK_B
Payload Contents:						

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	duration	ms	Duration of the requested task, set to zero for infinite duration. The maximum supported time is 12 days.
4	X4	-	flags	-	task flags (see graphic below)

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
backup	The receiver goes into backup mode for a time period defined by duration. Provided that it is not connected to USB

25.16.3.2 Requests a Power Management task

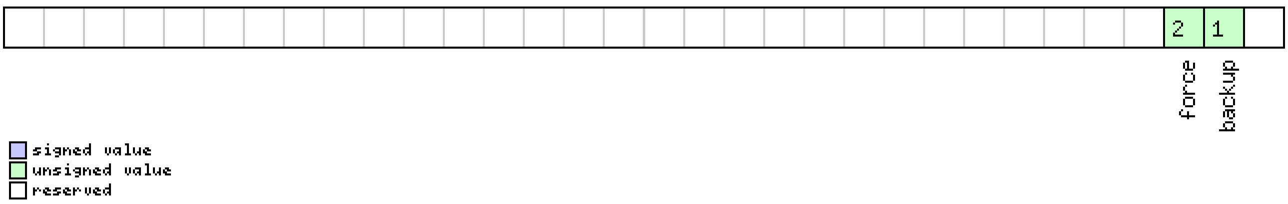
Message	**UBX-RXM-PMREQ**					
Description	**Requests a Power Management task**					
Type	Command					
Comment	Request of a Power Management related task of the receiver.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x02	0x41	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x00 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U4	-	duration	ms	Duration of the requested task, set to zero for infinite duration. The maximum supported time is 12 days.
8	X4	-	flags	-	task flags (see graphic below)
12	X4	-	wakeupSources	-	Configure pins to wakeup the receiver. The receiver wakes up if there is either a falling or a rising edge on one of the configured pins (see graphic below)

Bitfield flags

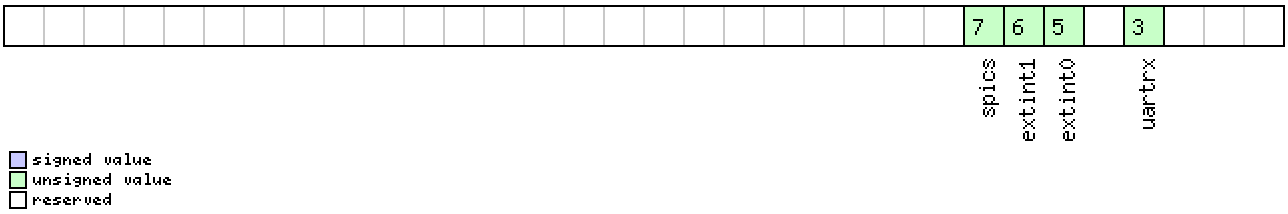
This graphic explains the bits of `flags`



Name	Description
backup	The receiver goes into backup mode for a time period defined by duration. Provided that it is not connected to USB
force	Force receiver backup while USB is connected. USB interface will be disabled.

Bitfield wakeupSources

This graphic explains the bits of wakeupSources



Name	Description
uartrx	Wakeup the receiver if there is an edge on the UART RX pin.
extint0	Wakeup the receiver if there is an edge on the EXTINT0 pin.
extint1	Wakeup the receiver if there is an edge on the EXTINT1 pin.
spics	Wakeup the receiver if there is an edge on the SPI CS pin.

25.16.4 UBX-RXM-RLM (0x02 0x59)

25.16.4.1 Galileo SAR Short-RLM report

Message	**UBX-RXM-RLM**				
Description	**Galileo SAR Short-RLM report**				
Type	Output				
Comment	This message contains the contents of any Galileo Search and Rescue (SAR) Short Return Link Message detected by the receiver.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x59	16	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x00 for this version)
1	U1	-	type	-	Message type (0x01 for Short-RLM)
2	U1	-	svId	-	Identifier of transmitting satellite (see Satellite Numbering)
3	U1	-	reserved1	-	Reserved
4	U1[8]	-	beacon	-	Beacon identifier (60 bits), with bytes ordered by earliest transmitted (most significant) first. Top four bits of first byte are zero.
12	U1	-	message	-	Message code (4 bits)
13	U1[2]	-	params	-	Parameters (16 bits), with bytes ordered by earliest transmitted (most significant) first.
15	U1	-	reserved2	-	Reserved

25.16.4.2 Galileo SAR Long-RLM report

Message	**UBX-RXM-RLM**						
Description	**Galileo SAR Long-RLM report**						
Type	Output						
Comment	This message contains the contents of any Galileo Search and Rescue (SAR) Long Return Link Message detected by the receiver.						
	Header	Class	ID	Length (Bytes)		Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x59	28		see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`version`	-	Message version (0x00 for this version)
1	U1	-	`type`	-	Message type (0x02 for Long-RLM)
2	U1	-	`svId`	-	Identifier of transmitting satellite (see Satellite Numbering)
3	U1	-	`reserved1`	-	Reserved
4	U1[8]	-	`beacon`	-	Beacon identifier (60 bits), with bytes ordered by earliest transmitted (most significant) first. Top four bits of first byte are zero.
12	U1	-	`message`	-	Message code (4 bits)
13	U1[12]	-	`params`	-	Parameters (96 bits), with bytes ordered by earliest transmitted (most significant) first.
25	U1[3]	-	`reserved2`	-	Reserved

25.16.5 UBX-RXM-SFRBX (0x02 0x13)

25.16.5.1 Broadcast Navigation Data Subframe

Message	**UBX-RXM-SFRBX**						
Description	**Broadcast Navigation Data Subframe**						
Type	Output						
Comment	This message reports a complete subframe of broadcast navigation data decoded from a single signal. The number of data words reported in each message depends on the nature of the signal. See the section on Broadcast Navigation Data for further details.						
	Header	Class	ID	Length (Bytes)		Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x13	8 + 4*numWords		see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`gnssId`	-	GNSS identifier (see Satellite Numbering)
1	U1	-	`svId`	-	Satellite identifier (see Satellite Numbering)
2	U1	-	`reserved1`	-	Reserved

UBX-RXM-SFRBX continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
3	U1	-	freqId	-	Only used for GLONASS: This is the frequency slot + 7 (range from 0 to 13)
4	U1	-	numWords	-	The number of data words contained in this message (up to 10, for currently supported signals)
5	U1	-	chn	-	The tracking channel number the message was received on
6	U1	-	version	-	Message version, (0x02 for this version)
7	U1	-	reserved2	-	Reserved
Start of repeated block (numWords times)					
8 + 4*N	U4	-	dwrd	-	The data words
End of repeated block					

25.16.6 UBX-RXM-SVSI (0x02 0x20)

25.16.6.1 SV Status Info

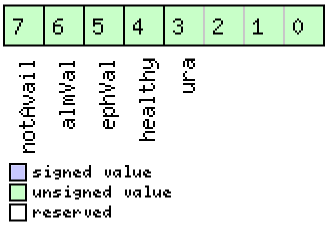
Message	**UBX-RXM-SVSI**				
Description	**SV Status Info**				
Type	Periodic/Polled				
Comment	Status of the receiver manager knowledge about GPS Orbit Validity This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-ORB message in preference.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x02	0x20	8 + 6*numSV	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch. See the description of iTOW for details.
4	I2	-	week	weeks	GPS week number of the navigation epoch
6	U1	-	numVis	-	Number of visible satellites
7	U1	-	numSV	-	Number of per-SV data blocks following
Start of repeated block (numSV times)					
8 + 6*N	U1	-	svid	-	Satellite ID
9 + 6*N	X1	-	svFlag	-	Information Flags (see graphic below)
10 + 6*N	I2	-	azim	-	Azimuth
12 + 6*N	I1	-	elev	-	Elevation
13 + 6*N	X1	-	age	-	Age of Almanac and Ephemeris: (see graphic below)
End of repeated block					

Bitfield svFlag

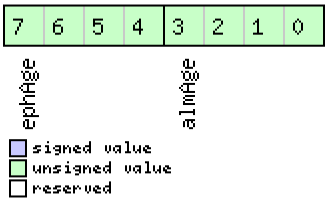
This graphic explains the bits of `svFlag`



Name	Description
ura	Figure of Merit (URA) range 0..15
healthy	SV healthy flag
ephVal	Ephemeris valid
almVal	Almanac valid
notAvail	SV not available

Bitfield age

This graphic explains the bits of `age`



Name	Description
almAge	Age of ALM in days offset by 4 i.e. the reference time may be in the future: ageOfAlm = (age & 0x0f) - 4
ephAge	Age of EPH in hours offset by 4. i.e. the reference time may be in the future: ageOfEph = ((age & 0xf0) >> 4) - 4

25.17 UBX-SEC (0x27)

Security Feature Messages

Messages in the SEC class are used for security features of the receiver.

25.17.1 UBX-SEC-SIGN (0x27 0x01)

25.17.1.1 Signature of a previous message

Message	**UBX-SEC-SIGN**					
Description	**Signature of a previous message**					
Type	Output					
Comment	The message is the signature of a previously sent message. The signature is generated with a hash using the SHA-256 algorithm with the programmed seeds.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x27	0x01	40	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x01 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U1	-	classID	-	Class ID of the referring message
5	U1	-	messageID	-	Message ID of the referring message
6	U2	-	checksum	-	UBX Checksum of the referring message
8	U1[32]	-	hash	-	SHA-256 hash of the referring message

25.17.2 UBX-SEC-UNIQID (0x27 0x03)

25.17.2.1 Unique Chip ID

Message	**UBX-SEC-UNIQID**					
Description	**Unique Chip ID**					
Type	Output					
Comment	This message is used to retrieve a unique chip identifier (40 bits, 5 bytes).					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x27	0x03	9	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	version	-	Message version (0x01 for this version)
1	U1[3]	-	reserved1	-	Reserved
4	U1[5]	-	uniqueId	-	Unique chip ID

25.18 UBX-TIM (0x0D)

Timing Messages: i.e. Time Pulse Output, Time Mark Results.
Messages in the TIM class are used to output timing information from the receiver, like Time Pulse and Time Mark measurements.

25.18.1 UBX-TIM-TM2 (0x0D 0x03)

25.18.1.1 Time mark data

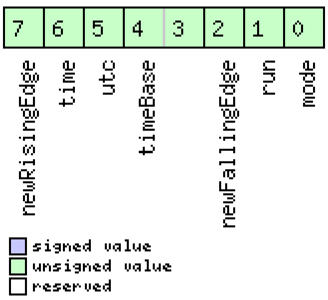
Message	**UBX-TIM-TM2**					
Description	**Time mark data**					
Type	Periodic/Polled					
Comment	This message contains information for high precision time stamping / pulse counting.					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0D	0x03	28	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	ch	-	Channel (i.e. EXTINT) upon which the pulse was measured
1	X1	-	flags	-	Bitmask (see graphic below)
2	U2	-	count	-	rising edge counter.
4	U2	-	wnR	-	week number of last rising edge
6	U2	-	wnF	-	week number of last falling edge
8	U4	-	towMsR	ms	tow of rising edge
12	U4	-	towSubMsR	ns	millisecond fraction of tow of rising edge in nanoseconds
16	U4	-	towMsF	ms	tow of falling edge
20	U4	-	towSubMsF	ns	millisecond fraction of tow of falling edge in nanoseconds
24	U4	-	accEst	ns	Accuracy estimate

Bitfield flags

This graphic explains the bits of `flags`



Name	Description
mode	0=single 1=running
run	0=armed 1=stopped
newFallingEdge	new falling edge detected
timeBase	0=Time base is Receiver Time 2=Time base is UTC (the variant according to the configuration in UBX-CFG-NAV5)
utc	0=UTC not available 1=UTC available
time	0=Time is not valid 1=Time is valid (Valid GNSS fix)
newRisingEdge	new rising edge detected

25.18.2 UBX-TIM-VRFY (0x0D 0x06)

25.18.2.1 Sourced Time Verification

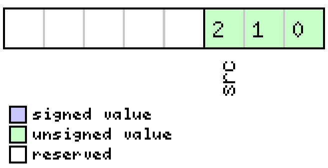
Message	UBX-TIM-VRFY					
Description	**Sourced Time Verification**					
Type	Periodic/Polled					
Comment	This message contains verification information about previous time received via AID-INI or from RTC					
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x0D	0x06	20	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	I4	-	itow	ms	integer millisecond tow received by source
4	I4	-	frac	ns	sub-millisecond part of tow
8	I4	-	deltaMs	ms	integer milliseconds of delta time (current time minus sourced time)
12	I4	-	deltaNs	ns	sub-millisecond part of delta time
16	U2	-	wno	week	week number
18	X1	-	flags	-	information flags (see graphic below)
19	U1	-	reserved1	-	Reserved

Bitfield flags

This graphic explains the bits of flags



Name	Description
src	aiding time source 0: no time aiding done 2: source was RTC 3: source was AID-INI

25.19 UBX-UPD (0x09)

Firmware Update Messages: i.e. Memory/Flash erase/write, Reboot, Flash identification, etc.. Messages in the UPD class are used to update the firmware and identify any attached flash device.

25.19.1 UBX-UPD-SOS (0x09 0x14)

25.19.1.1 Poll Backup File Restore Status

Message	**UBX-UPD-SOS**					
Description	**Poll Backup File Restore Status**					
Type	Poll Request					
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a System Restored from Backup message as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x09	0x14	0	see below	CK_A CK_B
No payload						

25.19.1.2 Create Backup File in Flash

Message	**UBX-UPD-SOS**					
Description	**Create Backup File in Flash**					
Type	Command					
Comment	The host can send this message in order to save part of the BBR memory in a file in flash file system. The feature is designed in order to emulate the presence of the backup battery even if it is not present; the host can issue the save on shutdown command before switching off the device supply. It is recommended to issue a GNSS stop command before, in order to keep the BBR memory content consistent.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x09	0x14	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	`cmd`	-	Command (must be 0)
1	U1[3]	-	`reserved1`	-	Reserved

25.19.1.3 Clear Backup in Flash

Message	**UBX-UPD-SOS**
Description	**Clear Backup in Flash**
Type	Command
Comment	The host can send this message in order to erase the backup file present in flash. It is recommended that the clear operation is issued after the host has received the notification that the memory has been restored after a reset. Alternatively the host can parse the startup string 'Restored data saved on shutdown' or poll the UBX-UPD-SOS message for getting the status.

	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x09	0x14	4	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	cmd	-	Command (must be 1)
1	U1[3]	-	reserved1	-	Reserved

25.19.1.4 Backup File Creation Acknowledge

Message	**UBX-UPD-SOS**
Description	**Backup File Creation Acknowledge**
Type	Output
Comment	The message is sent from the device as confirmation of creation of a backup file in flash. The host can safely shut down the device after received this message.

	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x09	0x14	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	cmd	-	Command (must be 2)
1	U1[3]	-	reserved1	-	Reserved
4	U1	-	response	-	0: Not acknowledged 1: Acknowledged
5	U1[3]	-	reserved2	-	Reserved

25.19.1.5 System Restored from Backup

Message	**UBX-UPD-SOS**				
Description	**System Restored from Backup**				
Type	Output				
Comment	The message is sent from the device to notify the host the BBR has been restored from a backup file in flash. The host should clear the backup file after receiving this message. If the UBX-UPD-SOS message is polled, this message will be resent.				
	Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure	0xB5 0x62	0x09	0x14	8	see below	CK_A CK_B

Payload Contents:

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	cmd	-	Command (must be 3)
1	U1[3]	-	reserved1	-	Reserved
4	U1	-	response	-	0: Unknown 1: Failed restoring from backup file 2: Restored from backup file 3: Not restored (no backup)
5	U1[3]	-	reserved2	-	Reserved

26 RTCM Protocol

The RTCM (Radio Technical Commission for Maritime Services) protocol is a protocol that is used to supply the GNSS receiver with real-time differential correction data. The RTCM protocol specification is available from http://www.rtcm.org.

26.1 RTCM2

26.1.1 Introduction

☞ This feature is only applicable to GPS operation.

☞ This feature only supports code differential positioning.

☞ For effective differential positioning accuracy, it is necessary that the reference station antenna is situated in a low multipath environment with an unobstructed view of the sky. It is recommended that reference receiver applies phase smoothing to the broadcast corrections.

☞ This feature is not available with the High Precision GNSS products.

26.1.2 Supported Messages

The following RTCM 2.3 messages are supported:

Supported RTCM 2.3 Message Types

Message Type	Description
1	Differential GPS Corrections
2	Delta Differential GPS Corrections
3	GPS Reference Station Parameters
9	GPS Partial Correction Set

26.1.3 Configuration

The DGPS feature does not need any configuration to work properly. When an RTCM stream is input on any of the communication interfaces, the data will be parsed and applied if possible, which will put the receiver into DGPS mode.

The only configurable parameter of DGPS mode is the timeout that can be specified using UBX-CFG-NAV5. This value defines the time after which old RTCM data will be discarded.

The RTCM protocol can be disabled/enabled on communication interfaces by means of the UBX-CFG-PRT message. By default, RTCM is enabled.

26.1.4 Output

DGPS mode will result in following modified output:

- NMEA-GGA: The quality field will be 2 (see NMEA Positon Fix Flags). The age of DGPS corrections and Reference station ID will be set.

- NMEA-GLL, NMEA-RMC, NMEA-VTG, NMEA-GNS: The posMode indicator will be D (see NMEA Positon Fix Flags).
- NMEA-PUBX-POSITION: The status will be D2/D3; The age of DGPS corrections will be set.
- UBX-NAV-SOL: The DGPS flag will be set.
- UBX-NAV-PVT: The diffSoln flag will be set.
- UBX-NAV-STATUS: The diffSoln flag will be set; the diffCorr flag will be set.
- UBX-NAV-SVINFO: The DGPS flag will be set for channels with valid DGPS correction data.
- UBX-NAV-DGPS: This message will contain all valid DGPS data
- If the base line exceeds 100km and a message type 3 is received, a UBX-INF-WARNING will be output, e.g. "WARNING: DGNSS baseline big: 330.3km"

26.1.5 Restrictions

The following restrictions apply to DGPS mode:

- The DGPS solution will only include measurements from satellites for which DGPS corrections were provided. This is because the navigation algorithms cannot mix corrected with uncorrected measurements.
- SBAS corrections will not be applied when using RTCM correction data.
- Precise Point Positioning will be deactivated when using RTCM correction data.
- RTCM correction data cannot be applied when using AssistNow Offline or AssistNow Autonomous.

26.1.6 Reference

The RTCM2 support is implemented according to RTCM 10402.3 ("RECOMMENDED STANDARDS FOR DIFFERENTIAL GNSS").

Appendix

A Satellite Numbering

A summary of all the SV numbering schemes is provided in the following table.

Satellite numbering

GNSS Type	SV range	UBX gnssId: svId	UBX svId	NMEA 2.X-4.0 (strict)	NMEA 2.X-4.0 (extended)	NMEA 4.10+ (strict)	NMEA 4.10+ (extended)
GPS	G1-G32	0:1-32	1-32	1-32	1-32	1-32	1-32
SBAS	S120-S158	1:120-158	120-158	33-64	33-64,152-158	33-64	33-64,152-158
Galileo	E1-E36	2:1-36	211-246	-	301-336	1-36	1-36
BeiDou	B1-B37	3:1-37	159-163,33-64	-	401-437	1-37	1-37
IMES	I1-I10	4:1-10	173-182	-	173-182	-	173-182
QZSS	Q1-Q5	5:1-5	193-197	-	193-197	-	193-197
GLONASS	R1-R32, R?	6:1-32, 6: 255	65-96, 255	65-96, null	65-96, null	65-96, null	65-96, null

B UBX and NMEA Signal Identifiers

UBX and NMEA protocols uses signal identifiers (commonly abbreviated to "sigId") to distinguish between different signals from GNSS.

Signal identifiers are only valid when combined with a GNSS identifier (see above). The table below shows the range of identifiers currently supported in the firmware.

C u-blox 8 / u-blox M8 Default Settings

The default settings listed in this section apply to u-blox 8 / u-blox M8 receivers. These values assume that the default levels of the configuration pins have been left unchanged and no setting that affects the default configuration was written to the eFuse. Default settings are dependent on the configuration pin and eFuse settings. For information regarding these settings, consult the applicable Data Sheet.

C.1 Antenna Supervisor Settings (UBX-CFG-ANT)

For parameter and protocol description see section UBX-CFG-ANT.

Antenna Supervisor Default Settings

Parameter	SPG 2.xx	SPG 3.xx, HPG 1.xx	ADR 3.xx	ADR 4.xx, UDR 1.xx	FTS 1.xx	TIM 1.0x	TIM 1.1x
flags-svcs	1	1	1	1	0	1	1
flags-scd	1	1	0	0	0	1	0
flags-pdwnOnSCD	1	1	0	0	0	0	0
flags-recovery	1	1	0	0	0	1	0
flags-ocd	0	0	0	0	0	0	0
pins-pinSwitch	16	16	16	16	31	16	16
pins-pinSCD	15	15	31	15	31	15	15
pins-pinOCD	31	14	31	14	31	31	14

C.2 Data Batching Settings (UBX-CFG-BATCH)

For parameter and protocol description see section UBX-CFG-BATCH.

Data Batching Default Settings

Parameter	SPG 3.51
flags-enable	0
flags-extraPvt	1
flags-extraOdo	1
flags-pioEnable	0
flags-pioActiveLow	0
bufSize	0
notifThrs	0
pioId	0

C.3 Datum Settings (UBX-CFG-DAT)

For parameter and protocol description see section UBX-CFG-DAT.

Datum Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
datumNum	0
datumName	WGS84
majA	6378137
flat	298.257223563
dX	0
dY	0
dZ	0
rotX	0
rotY	0
rotZ	0
scale	0

C.4 Geofencing Settings (UBX-CFG-GEOFENCE)

For parameter and protocol description see section UBX-CFG-GEOFENCE.

Geofencing Default Settings

Parameter	SPG 2.xx, SPG 3.xx, HPG 1.xx, ADR 3.xx, ADR 4.xx, UDR 1.xx
numFences	0
confLvl	0
pioEnabled	0
pinPolarity	0
pin	0

C.5 GNSS System Settings (UBX-CFG-GNSS)

For parameter and protocol description see section UBX-CFG-GNSS.

GNSS System Default Settings

Parameter	SPG 2.xx, ADR 3.xx	SPG 3.0x	ADR 4.xx, UDR 1.xx	FTS 1.xx	TIM 1.0x	TIM 1.1x, SPG 3.5x	HPG 1.xx
numTrkChHw	32	32	28	32	32	32	32
numTrkChUse	32	32	28	32	32	32	28
numConfigBlocks	5	7	7	5	6	7	4
gnssId	0, 1, 3, 5, 6	0, 1, 2, 3, 4, 5, 6	0, 1, 2, 3, 4, 5, 6	0, 1, 3, 5, 6	0, 1, 3, 4, 5, 6	0, 1, 2, 3, 4, 5, 6	0, 3, 5, 6
flags-enable	1, 1, 0, 1, 1	1, 1, 0, 0, 0, 1, 1	1, 1, 0, 0, 0, 1, 1	1, 0, 0, 1, 1	1, 0, 0, 0, 1, 1	1, 0, 0, 0, 0, 1, 1	1, 0, 1, 1
resTrkCh	8, 1, 8, 0, 8	8, 1, 4, 8, 0, 0, 8	8, 1, 4, 8, 0, 0, 8	8, 1, 8, 0, 8	8, 1, 8, 0, 0, 8	8, 1, 4, 8, 0, 0, 8	8, 8, 0, 8
maxTrkCh	16, 3, 16, 3, 14	16, 3, 8, 16, 8, 3, 14	16, 3, 8, 16, 8, 3, 14	16, 3, 16, 3, 14	16, 3, 16, 8, 3, 14	16, 3, 8, 16, 8, 3, 14	16, 16, 3, 14

C.6 INF Messages Settings (UBX-CFG-INF)

For parameter and protocol description see section UBX-CFG-INF.

C.6.1 UBX Protocol

INF Messages Default Settings for UBX protocol

Parameter	SPG 2.xx, SPG 3.xx, FTS 1.xx, TIM 1.xx, HPG 1.xx, ADR 3.xx, ADR 4.xx, UDR 1.xx
protocolID	0
infMsgMask-ERROR	0,0,0,0,0,0
infMsgMask-WARNING	0,0,0,0,0,0
infMsgMask-NOTICE	0,0,0,0,0,0
infMsgMask-TEST	0,0,0,0,0,0
infMsgMask-DEBUG	0,0,0,0,0,0

C.6.2 NMEA Protocol

INF Messages Default Settings for NMEA protocol

Parameter	SPG 2.xx, TIM 1.0x, FTS 1.xx, ADR 3.xx	SPG 3.xx, TIM 1.1x, HPG 1.xx	ADR 4.xx, UDR 1.xx
protocolID	1	1	1
infMsgMask-ERROR	1,1,1,1,1,1	1,1,0,1,1,0	1,1,0,1,1,0
infMsgMask-WARNING	1,1,1,1,1,1	1,1,0,1,1,0	1,1,0,1,1,0
infMsgMask-NOTICE	1,1,1,1,1,1	1,1,0,1,1,0	1,1,0,1,1,0
infMsgMask-TEST	0,0,0,0,0,0	0,0,0,0,0,0	0,0,0,0,0,0
infMsgMask-DEBUG	0,0,0,0,0,0	0,0,0,0,0,0	0,0,0,0,0,0

C.7 Jammer/Interference Monitor Settings (UBX-CFG-ITFM)

For parameter and protocol description see section UBX-CFG-ITFM.

Jamming/Interference Monitor Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
config-bbThreshold	3

Jamming/Interference Monitor Default Settings continued

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
config-cwThreshold	15
config-enable	0
config2-antSetting	0
config2-enable2	0

C.8 Logging Settings (UBX-CFG-LOGFILTER)

For parameter and protocol description see section UBX-CFG-LOGFILTER.

Logging Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
flags-recordEnabled	0
flags-psmOncePerWakupEnabled	0
flags-applyAllFilterSettings	0
minInterval	0
timeThreshold	0
speedThreshold	0
positionThreshold	0

C.9 Message Settings (UBX-CFG-MSG) (NOTE: unverified)

For parameter and protocol description see section UBX-CFG-MSG.

Enabled output messages

Message	Type	All Ports
NMEA-Standard-GGA	Out	1
NMEA-Standard-GLL	Out	1
NMEA-Standard-GSA	Out	1
NMEA-Standard-GSV	Out	1
NMEA-Standard-RMC	Out	1
NMEA-Standard-VTG	Out	1

C.10 Navigation Settings (UBX-CFG-NAV5)

For parameter and protocol description see section UBX-CFG-NAV5.

Navigation Default Settings

Parameter	SPG 2.xx, ADR 3.xx	SPG 3.xx	ADR 4.xx, UDR 1.xx	FTS 1.xx	TIM 1.0x	TIM 1.1x	HPG 1.xx
mask-dyn	1	1	1	1	1	1	1
mask-minEl	1	1	1	1	1	1	1
mask-posFixMode	1	1	1	1	1	1	1
mask-drLim	1	1	1	1	1	1	1
mask-posMask	1	1	1	1	1	1	1
mask-timeMask	1	1	1	1	1	1	1
mask-staticHoldMask	1	1	1	1	1	1	1

Navigation Default Settings continued

Parameter	SPG 2.xx, ADR 3.xx	SPG 3.xx	ADR 4.xx, UDR 1.xx	FTS 1.xx	TIM 1.0x	TIM 1.1x	HPG 1.xx
mask-dgpsMask	1	1	1	1	1	1	1
mask-cnoThreshold	1	1	1	1	1	1	1
mask-utc	1	1	1	1	1	1	1
dynModel	0	0	4	2	2	2	0
fixMode	3	3	3	3	3	3	3
fixedAlt	0	0	0	0	0	0	0
fixedAltVar	1	1	1	1	1	1	1
minElev	5	5	10	5	5	5	10
drLimit	0	0	0	0	0	0	0
pDop	25	25	25	25	25	25	25
tDop	25	25	25	25	25	25	25
pAcc	100	100	100	100	100	100	100
tAcc	300	350	350	300	350	350	350
staticHoldThresh	0	0	0	0	0	0	0
dgpsTimeOut	60	60	60	60	60	60	60
cnoThreshNumSVs	0	0	0	0	0	0	0
cnoThresh	0	0	0	0	0	0	0
staticHoldMaxDist	200	0	0	200	200	0	0
utcStandard	0	0	0	3	3	3	0

C.11 Navigation Settings (UBX-CFG-NAVX5)

For parameter and protocol description see section UBX-CFG-NAVX5.

Navigation Default Settings (SPG/FTS/TIM)

Parameter	SPG 2.xx	SPG 3.0x	SPG 3.5x	FTS 1.xx, TIM 1.0x	TIM 1.1x
mask1-minMax	1	1	1	1	1
mask1-minCno	1	1	1	1	1
mask1-initial3dfix	1	1	1	1	1
mask1-wknRoll	1	1	1	1	1
mask1-ackAid	1	1	1	1	1
mask1-ppp	1	1	1	1	1
mask1-aop	1	1	1	1	1
mask2-adr	0	0	0	0	0
minSVs	3	3	3	1	1
maxSVs	20	32	32	20	32
minCNO	6	6	6	9	9
iniFix3D	0	0	0	0	0
ackAiding	0	0	0	0	0
wknRollover	1756	1867	1936	1756	1867
usePPP	0	0	0	0	0
aopCfg-useAOP	0	0	0	0	0
aopOrbMaxErr	100	100	100	100	100
gnssTofsCfg-tolerance	0	0	0	0	0

Navigation Default Settings (SPG/FTS/TIM) continued

Parameter	SPG 2.xx	SPG 3.0x	SPG 3.5x	FTS 1.xx, TIM 1.0x	TIM 1.1x
gnssTofsCfg-useMeasVarTest	0	0	0	0	0
gnssTofsCfg-aopPreCalEnabled	0	0	0	0	0
gnssTofsCfg-aopPreCalDt	0	0	0	0	0
gnssTofsCfg-aopPreCalInhInt	0	0	0	0	0
useAdr	0	0	0	0	0

C.12 NMEA Protocol Settings (UBX-CFG-NMEA)

For parameter and protocol description see section UBX-CFG-NMEA.

NMEA Protocol Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
filter-posFilt	0
filter-mskPosFilt	0
filter-timeFilt	0
filter-dateFilt	0
filter-gpsOnlyFilter	0
filter-trackFilt	0
nmeaVersion	0x40
numSV	0
flags-compat	0
flags-consider	1
flags-limit82	0
flags-highPrec	0
gnssToFilter-gps	0
gnssToFilter-sbas	0
gnssToFilter-qzss	0
gnssToFilter-glonass	0
gnssToFilter-beidou	0
svNumbering	0
mainTalkerId	0
gsvTalkerId	0
bdsTalkerId	not set

C.13 Odometer Settings (UBX-CFG-ODO)

For parameter and protocol description see section UBX-CFG-ODO.

ODO Default Settings

Parameter	SPG 2.xx, SPG 3.0x, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx	SPG 3.5x
flags-useODO	0	1
flags-useCOG	0	1

ODO Default Settings continued

Parameter	SPG 2.xx, SPG 3.0x, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx	SPG 3.5x
flags-outLPVel	0	1
flags-outLPCog	0	1
odoCfg-profile	0	0
cogMaxSpeed	1	1
cogMaxPosAcc	50	50
velLpGain	153	153
cogLpGain	76	76

C.14 Power Management 2 Configuration (UBX-CFG-PM2)

For parameter and protocol description see section UBX-CFG-PM2.

Power Management 2 Configuration Default Settings

Parameter	SPG 2.xx, ADR 3.xx, FTS 1.xx, ADR 4.xx, UDR 1.xx	SPG 3.0x	SPG 3.51	TIM 1.0x	TIM 1.1x
maxStartupStateDur	0	0	0	0	0
flags-extintSel	0	0	0	0	0
flags-extintWake	0	0	0	0	0
flags-extintBackup	0	0	0	0	0
flags-extintInactive	n/a	0	0	n/a	0
flags-limitPeakCurr	0	0	0	0	0
flags-waitTimeFix	0	0	0	1	1
flags-updateRTC	0	0	0	0	0
flags-updateEPH	1	1	0	1	1
flags-doNotEnterOff	0	0	1	0	0
flags-mode	1	1	1	1	1
updatePeriod	1000	1000	1000	1000	1000
searchPeriod	10000	10000	10000	10000	10000
gridOffset	0	0	0	0	0
onTime	0	0	0	0	0
minAcqTime	0	0	300	0	0
extintInactivityMs	n/a	0	0	n/a	0

C.15 Port Configuration (UBX-CFG-PRT)

For parameter and protocol description see section UBX-CFG-PRT.

C.15.1 UART Port Configuration

For parameter and protocol description see section UBX-CFG-PRT-UART.

UART 1 Default Settings

Parameter	SPG 2.xx, SPG 3.xx, FTS 1.xx, TIM 1.xx	ADR 3.xx, ADR 4.xx, UDR 1.xx	HPG 1.xx
txReady-en	0	0	0

UART 1 Default Settings continued

Parameter	SPG 2.xx, SPG 3.xx, FTS 1.xx, TIM 1.xx	ADR 3.xx, ADR 4.xx, UDR 1.xx	HPG 1.xx
txReady-pol	0	0	0
txReady-pin	0	0	0
txReady-thres	0	0	0
baudRate	9600	9600	9600
inProtoMask	inUbx,inNmea,inRtcm	inUbx,inNmea,inRtcm	inUbx,inNmea, inRtcm3
outProtoMask	outUbx,outNmea	outUbx,outNmea	outUbx,outNmea, outRtcm3
flags-extendedTxTimeout	0	0	0

C.15.2 SPI Port Configuration

For parameter and protocol description see section UBX-CFG-PRT-SPI.

SPI Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
txReady-en	0
txReady-pol	0
txReady-pin	0
txReady-thres	0
mode-spiMode	0
mode-flowControl	0
mode-ffCnt	0
inProtoMask	None
outProtoMask	None
flags-extendedTxTimeout	0

C.15.3 DDC Port Configuration

For parameter and protocol description see section UBX-CFG-PRT-DDC.

DDC Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx	HPG 1.xx
txReady-en	0	0
txReady-pol	0	0
txReady-pin	0	0
txReady-thres	0	0
mode-slaveAddr	0x42	0x42
inProtoMask	inUbx,inNmea,inRtcm	inUbx,inNmea,inRtcm3
outProtoMask	outUbx,outNmea	outUbx,outNmea,outRtcm3
flags-extendedTxTimeout	0	0

C.16 Output Rate Settings (UBX-CFG-RATE)

For parameter and protocol description see section UBX-CFG-RATE.

Output Rate Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, ADR 4.xx, UDR 1.xx, HPG 1.xx
measRate	1000
navRate	1
timeRef	1

C.17 Remote Inventory Settings (UBX-CFG-RINV)

For parameter and protocol description see section UBX-CFG-RINV.

Remote Inventory Default Settings

Parameter	SPG 2.xx, SPG 3.xx, ADR 3.xx, FTS 1.xx, TIM 1.xx, HPG 1.xx
flags-dump	0
flags-binary	0

C.18 Receiver Manager Configuration Settings (UBX-CFG-RXM)

For parameter and protocol description see section UBX-CFG-RXM.

Power Management Default Settings

Parameter	SPG 2.xx, FTS 1.xx, TIM 1.0x	SPG 3.0x, TIM 1.1x, HPG 1.xx	ADR 3.xx	ADR 4.xx, UDR 1.xx	SPG 3.5x
lpMode	0	0	0	0	1

C.19 SBAS Configuration Settings (UBX-CFG-SBAS)

For parameter and protocol description see section UBX-CFG-SBAS.

SBAS Configuration Default Settings

Parameter	SPG 2.xx, FTS 1.xx, TIM 1.0x	SPG 3.0x	SPG 3.5x	ADR 3.xx	ADR 4.xx, UDR 1.xx	TIM 1.1x
mode-enabled *	1	1	1	1	1	0
mode-test	0	0	0	0	0	0
usage-range	1	1	1	1	1	1
usage-diffCorr	1	1	1	1	1	1
usage-integrity	0	0	0	0	0	0
maxSBAS *	3	3	3	3	3	3
scanmode2	None	None	None	None	None	None
scanmode1	120,124, 126,129, 133,135, 137,138	120,123, 127-129, 133,135-138	120,123, 127-129, 133,135-138	120,124, 126,127-129,133, 135,137, 138	120,123, 127-129, 133,135-138	120,123, 127-129, 133,135-138

* These parameters are deprecated; use UBX-CFG-GNSS instead.

Related Documents

Overview

As part of our commitment to customer support, u-blox maintains an extensive volume of technical documentation for our products. In addition to product-specific data sheets and integration manuals, general documents are also available. These include:

- GPS Compendium, Docu. No GPS-X-02007
- GPS Antennas - RF Design Considerations for u-blox GPS Receivers, Docu. No GPS-X-08014

Our website www.u-blox.com is a valuable resource for general and product specific documentation.

For design and integration projects the Receiver Description Including Interface Description should be used together with the Data Sheet and Hardware Integration Manual of the GNSS receiver.

Revision History

Revision	Date	Name	Status / Comments
R01	30-Aug-2019	jesk	u-blox ZED-M8B specific description

Contact

For complete contact information visit us at www.u-blox.com

u-blox Offices

North, Central and South America

u-blox America, Inc.
Phone: +1 703 483 3180
E-mail: info_us@u-blox.com

Regional Office West Coast:
Phone: +1 408 573 3640
E-mail: info_us@u-blox.com

Technical Support:
Phone: +1 703 483 3185
E-mail: support_us@u-blox.com

Headquarters
Europe, Middle East, Africa

u-blox AG
Phone: +41 44 722 74 44
E-mail: info@u-blox.com
Support: support@u-blox.com

Asia, Australia, Pacific

u-blox Singapore Pte. Ltd.
Phone: +65 6734 3811
E-mail: info_ap@u-blox.com
Support: support_ap@u-blox.com

Regional Office Australia:
Phone: +61 2 8448 2016
E-mail: info_anz@u-blox.com
Support: support_ap@u-blox.com

Regional Office China (Beijing):
Phone: +86 10 68 133 545
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office China (Chongqing):
Phone: +86 23 6815 1588
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office China (Shanghai):
Phone: +86 21 6090 4832
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office China (Shenzhen):
Phone: +86 755 8627 1083
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office India:
Phone: +91 80 4050 9200
E-mail: info_in@u-blox.com
Support: support_in@u-blox.com

Regional Office Japan (Osaka):
Phone: +81 6 6941 3660
E-mail: info_jp@u-blox.com
Support: support_jp@u-blox.com

Regional Office Japan (Tokyo):
Phone: +81 3 5775 3850
E-mail: info_jp@u-blox.com
Support: support_jp@u-blox.com

Regional Office Korea:
Phone: +82 2 542 0861
E-mail: info_kr@u-blox.com
Support: support_kr@u-blox.com

Regional Office Taiwan:
Phone: +886 2 2657 1090
E-mail: info_tw@u-blox.com
Support: support_tw@u-blox.com