GPS

locate, communicate, accelerate

u-blox 6

Receiver Description

Including Protocol Specification

Abstract

The Receiver Description Including Protocol Specification describes the firmware features, specifications and configuration for u-blox 6 high performance GPS receivers. u-blox 6 firmware includes many features and configuration settings to customize receiver behavior to the user's specific needs.
The Receiver Description provides an overview and conceptual details of the supported features. The Protocol Specification details the NMEA and UBX protocols and serves as a reference tool.

Document Information	
Title	u-blox 6 Receiver Description
Subtitle	Including Protocol Specification
Document type	Manual
Document number	GPS.G6-SW-10018-F 68000, 18 Apr 2013
Document status	Revision for FW 7.03 (Public Release)

Table of Contents

Receiver Description

1 Overview

The Receiver Description including Protocol Specification is an important resource for integrating and configuring your u-blox 6 GPS receiver. This document has a modular structure and it is not necessary to read it from the beginning to the end. There are 2 main sections: The Receiver Description and the Protocol Specification.

The Receiver Description describes the software aspects of system features and configuration of u-blox 6 GPS technology. The Receiver Description is structured according to functionalities, with links provided to the corresponding NMEA and UBX messages, which are described in the Protocol Specification.

The Protocol Specification is a reference describing the software messages used by your u-blox receiver and is organized by the specific NMEA and UBX messages.

This document provides general information on the u-blox 6 GPS receiver **firmware**. *Some information might not apply to certain products that use said firmware. Refer to the product data sheet and/or the hardware integration manual for possible restrictions.*

2 Navigation Configuration Settings Description

This section relates to the configuration message CFG-NAV5.

2.1 Platform settings

u-blox positioning technology supports different dynamic platform models 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 results in a loss of receiver performance and position accuracy.

Dynamic Platform Model

Platform	*Description*
Portable	Default setting. Applications with low acceleration, e.g. portable devices. Suitable for most situations. MAX Altitude [m]: 12000, MAX Velocity [m/s]: 310, MAX Vertical Velocity [m/s]: 50, Sanity check type: Altitude and Velocity, Max Position Deviation: Medium
Stationary	Used in timing applications (antenna must be stationary) or other stationary applications. Velocity restricted to 0 m/s. Zero dynamics assumed. MAX Altitude [m]: 9000, MAX Velocity [m/s]: 10, MAX Vertical Velocity [m/s]: 6, Sanity check type: Altitude and Velocity, Max Position Deviation: Small
Pedestrian	Applications with low acceleration and speed, e.g. how a pedestrian would move. Low acceleration assumed. MAX Altitude [m]: 9000, MAX Velocity [m/s]: 30, MAX Vertical Velocity [m/s]: 20, Sanity check type: Altitude and Velocity, Max Position Deviation: Small
Automotive	Default setting for ADR. Used for applications with equivalent dynamics to those of a passenger car. Low vertical acceleration assumed. MAX Altitude [m]: 6000 (5000 for firmware versions 6.00 and below), MAX Velocity [m/s]: 84 (62 for firmware versions 4.00 to 5.00), MAX Vertical Velocity [m/s]: 15, Sanity check type: Altitude and Velocity, Max Position Deviation: Medium
At sea	Recommended for applications at sea, with zero vertical velocity. Zero vertical velocity assumed. Sea level assumed. MAX Altitude [m]: 500, MAX Velocity [m/s]: 25, MAX Vertical Velocity [m/s]: 5, Sanity check type: Altitude and Velocity, Max Position Deviation: Medium

Dynamic Platform Model continued

Platform	*Description*
Airborne <1g	Used for applications with a higher dynamic range and vertical acceleration than a passenger car. No 2D position fixes supported. MAX Altitude [m]: 50000, MAX Velocity [m/s]: 100, MAX Vertical Velocity [m/s]: 100, Sanity check type: Altitude, Max Position Deviation: Large
Airborne <2g	Recommended for typical airborne environment. No 2D position fixes supported. MAX Altitude [m]: 50000, MAX Velocity [m/s]: 250, MAX Vertical Velocity [m/s]: 100, Sanity check type: Altitude, Max Position Deviation: Large
Airborne <4g	Only recommended for extremely dynamic environments. No 2D position fixes supported. MAX Altitude [m]: 50000, MAX Velocity [m/s]: 500, MAX Vertical Velocity [m/s]: 100, Sanity check type: Altitude, Max Position Deviation: Large

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

2.2 Navigation Input Filters

The navigation input filters in 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 permanently 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.
drLimit	Dead reckoning limit: The time during which the receiver provides an extrapolated solution. After the DR timeout has expired, no position solution is provided.

See also comments in section Degraded Navigation below.

2.3 Navigation Output Filters

The navigation output filters in CFG-NAV5 adjust the valid flag of the relevant NMEA and UBX output messages. Users of the UBX protocol have additional access to messages containing an accuracy indicator, along with the position, time and velocity solutions.

- The **pDop** and **pAcc** values: The PDOP and Position Accuracy Mask are used to determine if a position solution is marked valid in the NMEA sentences or if the UBX gpsFixOk flag is set (UBX-NAV-STATUS and UBX-NAV-SOL). A solution is considered valid, when both PDOP and Accuracy lie below the respective limits.
- The **tDop** and **tAcc** values: The TDOP and Time Accuracy Mask are used to determine when a time pulse should be allowed. The time pulse is disabled if either TDOP or the time accuracy exceeds its respective limit. See also the TIM-TP message description.

Important: To qualify a position as valid the gpsFixOK flag in the **UBX-NAV-STATUS** *message must be checked. gpsFix=3D/3D in the* **UBX-NAV-STATUS** *message does not qualify a fix as valid and*

within the limits. To qualify a position as valid and within the pDop and pAcc limits set in the **UBX-CFG-NAV5** *message the gpsFixOK flag in the* **UBX-NAV-STATUS** *message has to be checked.*

Important: To qualify the speed information as valid the gpsFixOK flag in the **UBX-NAV-STATUS** *message must be checked.*

2.4 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.

Static Hold mode may not be used on GPS receivers with Automotive Dead Reckoning (ADR) enabled.

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.

2.5 Freezing the Course Over Ground

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).

The course over ground will never be frozen on GPS receivers with Automotive Dead Reckoning (ADR) enabled.

2.6 Degraded Navigation

Degraded navigation describes all navigation modes which use less than 4 Satellite Vehicles (SVs).

2.6.1 2D Navigation

If the receiver only has 3 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. 4 SVs available), the altitude is kept constant at the last known value. This is called a 2D fix.

u-blox positioning technology does not calculate any solution with less than 3 SVs. Only u-blox timing receivers can, when stationary, calculate a timing solution with only 1 SV.

2.6.2 Dead Reckoning, Extrapolating Positioning

This linear extrapolation feature is enabled by setting the `drLimit` parameter in CFG-NAV5. The extrapolation algorithm becomes active as soon as the receiver no longer achieves a position fix with a sufficient position accuracy or DOP value (see section Navigation Output Filters). It keeps a fixed track (heading is equal to the last calculated heading) until the dead reckoning limit is reached, or a position fix is again possible. The position is extrapolated, and the fix type is indicated as 1 (DR only). See NMEA V2.1 for NMEA fix flags.

For automotive dead reckoning (ADR), u-blox offers a solution based on input from external sensors as

described in section Description of Automotive Dead Reckoning (ADR). The mentioned ADR solution is unrelated to this linear extrapolation feature. The ADR solution allows high accuracy position solutions for automotive applications in situations with poor or no GPS coverage. This technology relies on additional inputs such as a turn rate sensor (gyro) or a speed sensor (odometer or wheel tick).

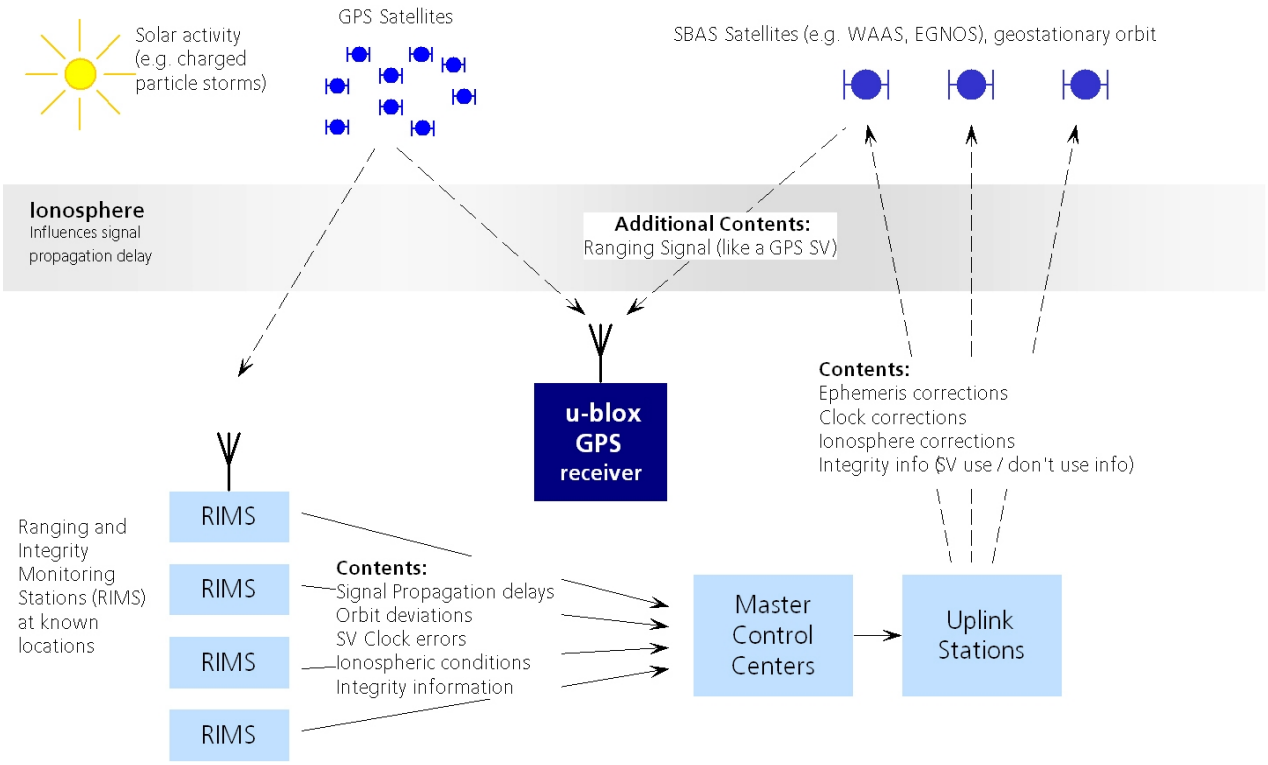
Do not use the linear extrapolation feature together with a u-blox ADR sensor-based Dead Reckoning GPS solution, as it will dilute the result!

3 SBAS Configuration Settings Description

3.1 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 (GEOs) 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.

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 Asia has been in operation since 2007.
- EGNOS (European Geostationary Navigation Overlay Service) is at the time of writing in test mode.
- GAGAN (GPS Aided Geo Augmented Navigation), developed by the Indian government is at the time of writing in test mode.

SBAS support allows u-blox GPS technology to take full advantage of the augmentation systems that are currently available (WAAS, EGNOS, MSAS), as well as those being tested and planned (such as GAGAN).

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 please refer to

- 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.

GEO satellites used by WAAS, EGNOS and MSAS (as of November 2010)

GEO Identification	*Position*	*GPS PRN*	*SBAS Provider*
AMR	98° W	133	WAAS
Inmarsat 3F3, POR	178° E	134	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° E	129	MSAS
MTSAT-2	145° E	137	MSAS

3.2 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 satellites in parallel, even from different SBAS systems (WAAS, EGNOS, MSAS, etc.). They can be tracked and used for navigation simultaneously. At least three SBAS satellites can be tracked in parallel. Every SBAS satellite tracked utilizes one vacant GPS receiver tracking channel. Only the number of receiver channels limits the total number of satellites used. Each SBAS satellite, which broadcasts ephemeris or almanac information, can be used for navigation, just like a normal GPS satellite.

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

In case corrections are available from the chosen GEO and used in the navigation calculation, the DGPS flag is set in the receiver's output protocol messages (see `NAV-SOL`, `NAV-STATUS`, `NAV-SVINFO`, `NMEA Position Fix Flags description`). The message `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 RIMS stations of a region are combined to a TEC (Total Electron Content) Map. This map is transferred to the GPS devices via the GEOs to allow a correction of the ionosphere error on each received satellite.

Supported SBAS messages

Message Type	*Message Content*	*Used from GEO*
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	GEO Navigation (Ephemeris)	All
10	Degradation	Primary
12	Time Offset	Primary
17	GEO Almanacs	All
18	Ionosphere Grid Point Assignment	Primary
24	Mixed Fast / Long term Corrections	Primary
25	Long term Corrections	Primary
26	Ionosphere Delays	Primary

As each GEO services a specific region, the correction signal is only useful within that region. Therefore, mission planning is crucial to determine the best possible configuration. The different stages (Testmode vs. Operational) of the various SBAS systems further complicate this task. The following examples show possible scenarios:

Example 1: SBAS Receiver in North America

At the time of writing, the WAAS system is in operational stage, whereas the EGNOS system is still in test mode. Therefore, and especially in the eastern parts of the US, care must be taken in order not to have EGNOS satellites taking preference over WAAS satellites. This can be achieved by disallowing Test Mode use (this inhibits EGNOS satellites from being used as a correction data source), but keeping the PRN Mask to have all SBAS GEOs enabled (which allows EGNOS GEOs to be used for navigation).

Example 2: SBAS Receiver in Europe

At the time of writing, the EGNOS system is still in test mode. To try out EGNOS operation, Testmode usage must be enabled. Since some WAAS satellites can be received in the western parts of Europe but don't carry correction data for the European continent, the GEOs from all but the EGNOS system should be disallowed, using the PRN Mask. It is important to understand that while EGNOS is in test mode, anything can happen to the EGNOS signals, such as sudden interruption of service or broadcast of invalid or inconsistent data.

Although u-blox GPS receivers try to select the best available SBAS correction data, it is recommended to disallow the usage of unwanted SBAS satellites by configuration.

3.3 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	Enables or disables the SBAS subsystem
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

SBAS Configuration parameters continued

Parameter	*Description*
Services/Usage - Apply integrity information	Use integrity data
Number of tracking channels	Sets how many channels are reserved for SBAS tracking (e.g., if this is set to three and five SBAS SVs are acquired, only three of these will prioritized over available GPS signals.
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.

4 Serial Communication Ports Description

u-blox positioning technology comes 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 CFG-PRT. This message also allows changing port-specific settings (baud rate, address etc.). See `CFG-MSG` for a description of the mechanism for enabling and disabling messages.

A `target` in the context of the I/O system is an I/O port. The following table shows the target numbers used

Target Number assignment

Target #	*Electrical Interface*
0	DDC (I2C compatible)
1	UART 1
2	UART 2
3	USB
4	SPI
5	reserved

4.1 UART Ports

One or two Universal Asynchronous Receiver/Transmitter (UART) ports are featured, that can be used to transmit GPS measurements, monitor status information and configure the receiver. See our online product selector matrix 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 or for different protocols on the same port.

Possible UART Interface Configurations

Baud Rate	*Data Bits*	*Parity*	*Stop Bits*
4800	8	none	1
9600	8	none	1
19200	8	none	1

Possible UART Interface Configurations continued

Baud Rate	Data Bits	Parity	Stop Bits
38400	8	none	1
57600	8	none	1
115200	8	none	1

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 ensure data validity on all communication interfaces (SPI, DDC, USB, UART) Firmware 6.02 implements a maximum lifetime for transmit packets of 2 seconds. After a message is generated, if transmission does not begin within this time limit, the message will be discarded.

If the number of bytes to be transmitted and the baud rate are selected so that transmission cannot be fully completed within the timeout period, then the host will not receive some messages. To prevent message losses due to timeout, the baudrate 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.

Firmware 7.01 and later do not implement a timeout for messages, but will drop new messages if the internal buffer is full.

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 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.

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.

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

4.2 USB Port

One Universal Serial Bus (USB) port is featured. See our online product selector matrix for availability. This port can be used for communication purposes and to power the GPS receiver.

The USB interface supports two different power modes:

- In *Self Powered Mode* the receiver is powered by its own power supply. **VDDUSB** is used to detect the availability of the USB port, i.e. whether the receiver is connected to a USB host.
- In *Bus Powered Mode* the device is powered by the USB bus, therefore no additional power supply is needed. In this mode the default maximum current that can be drawn by the receiver is 100 mA for u-blox 6 (120 mA for u-blox 5). See CFG-USB for a description on how to change this maximum. Configuring Bus Powered Mode indicates that the device will enter a low power state with disabled GPS functionality when the host suspends the device, e.g. when the host is put into stand-by mode.

The voltage range for **VDDUSB** *is specified from 3.0V to 3.6V, which differs slightly from the specification for VCC*

4.3 DDC Port

A Display Data Channel (DDC) bus is implemented, which is a 2-wire communication interface compatible with the I2C standard (Inter-Integrated Circuit). See our online product selector matrix for availability.

Unlike all other interfaces, the DDC is not able to communicate in full-duplex mode, i.e. TX and RX are mutually

exclusive. u-blox receivers act as a slave in the communication setup, therefore they cannot initiate data transfers on their own. The host, which is always master, provides the data clock (SCL), and the clock frequency is therefore not configurable on the slave.

The clock rate on the SCL line generated by the master must not exceed 100kHz (standard-mode).

The receiver's DDC address is set to 0x42 by default. This address can be changed by setting the **mode** field in CFG-PRT for DDC accordingly.

As the receiver will be run in slave mode and the physical layer lacks a handshake mechanism to inform the master about data availability, a layer has been inserted between the physical layer and the UBX and NMEA layer. The DDC implements a simple streaming interface that allows the constant polling of data, discarding everything that is not parseable. This means that the receiver returns 0xFF if no data is available. With firmware 7.01 the TX-ready feature was introduced to inform the master about data availability. It can be used as a trigger for data transmission.

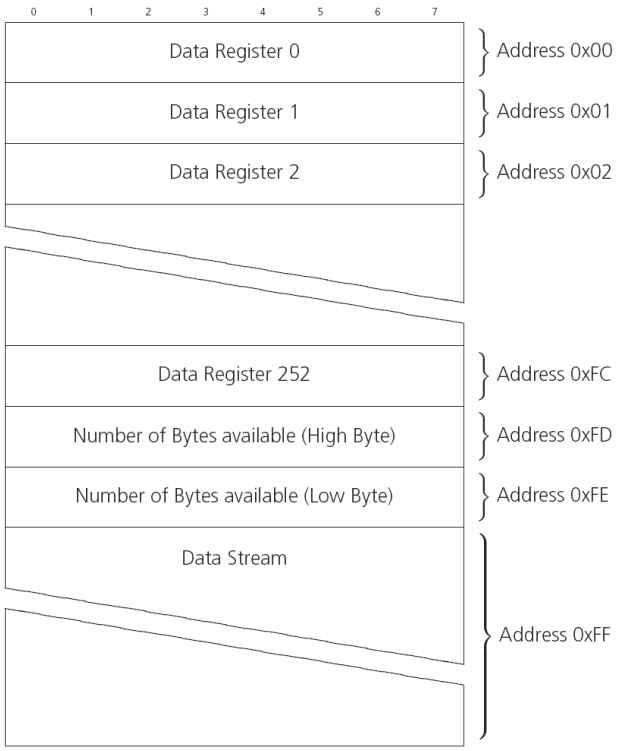
If no data is polled for 2 seconds, the interface is assumed to be idle. The receiver clears all pending data and no new messages will be scheduled to this interface. This mechanism can be disabled using the extended TX timeout flag in the port configuration, which allows longer time without bus read access. Note that interface data will be deleted when the internal buffer limit of 4 kB is exceeded.

4.3.1 Read Access

To allow both polled access to the full message stream and quick access to the key data, the register layout depicted in Figure *DDC Register Layout* is provided. The data registers 0 to 252, at addresses 0x00 to 0xFC, each 1 byte in size, contain information to be defined at a later point in time. At addresses 0xFD and 0xFE, the currently available number of bytes in the message stream can be read. At address 0xFF, the message stream is located. Subsequent reads from 0xFF return the messages in the transmit buffer, byte by byte. If the number of bytes read exceeds the number of bytes indicated, the payload is padded using the value 0xFF.

The registers 0x00 to 0xFC will be defined in a later firmware release. Do not use them, as they don't provide any meaningful data!

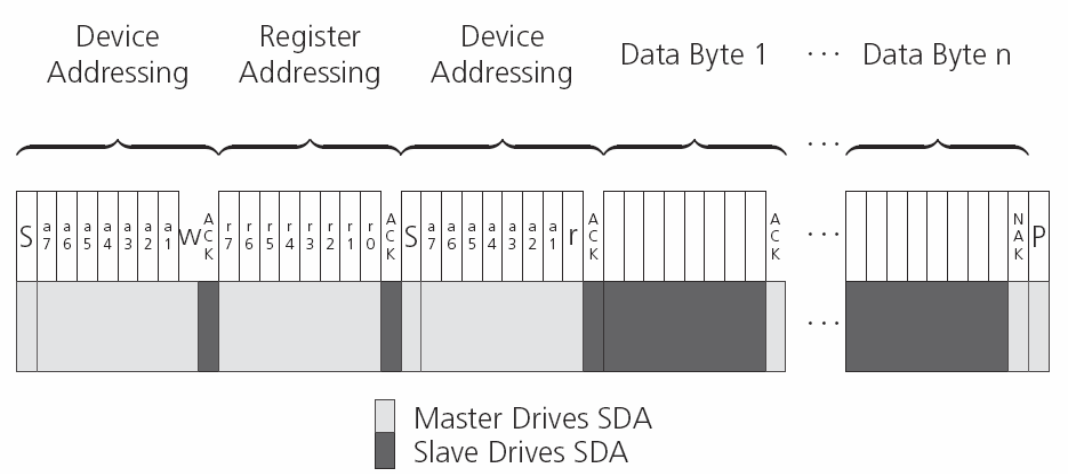
DDC Register Layout



4.3.1.1 Random Read Access

Random read operations allow the master to access any register in a random manner. To perform this type of read operation, first the register address to read from must be written to the receiver (see Figure *DDC Random Read Access*). Following the start condition from the master, the 7-bit device address and the `RW` bit (which is a logic low for write access) are clocked onto the bus by the master transmitter. The receiver answers with an acknowledge (logic low) to indicate that it is responsible for the given address. Next, the 8-bit address of the register to be read must be written to the bus. Following the receiver's acknowledge, the master again triggers a start condition and writes the device address, but this time the `RW` bit is a logic high to initiate the read access. Now, the master can read 1 to `N` bytes from the receiver, generating a not-acknowledge and a stop condition after the last byte being read. After every byte being read, the internal address counter is incremented by one, saturating at 0xFF. This saturation means, that, after having read all registers coming after the initially set register address, the raw message stream can be read.

DDC Random Read Access

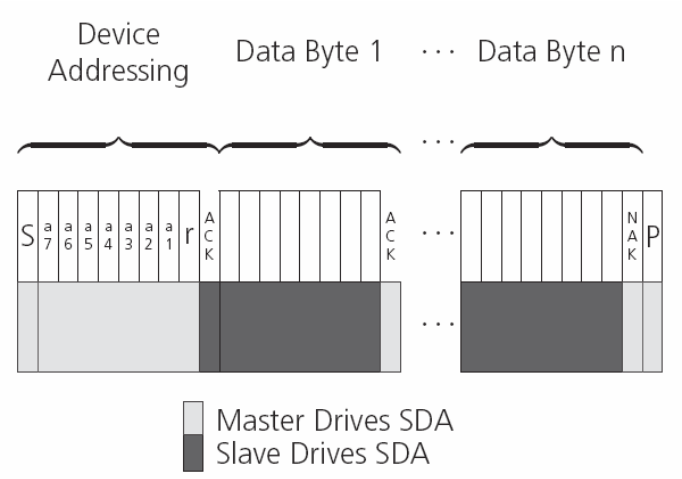


4.3.1.2 Current Address Read

The receiver contains an address counter that maintains the address of the last register accessed, internally incremented by one. Therefore, if the previous read access was to address `n` (where `n` is any legal address), the next current address read operation would access data from address `n+1` (see Figure *DDC Current Address Read Access*). Upon receipt of the device address with the `RW` bit set to one, the receiver issues an acknowledge and the master can read `1` to `N` bytes from the receiver, generating a not-acknowledge and a stop condition after the last byte being read.

To allow direct access to streaming data, the internal address counter is initialized to 0xFF, meaning that current address reads without a preceding random read access return the raw message stream. The address counter can be set to another address at any point using a random read access.

DDC Current Address Read Access

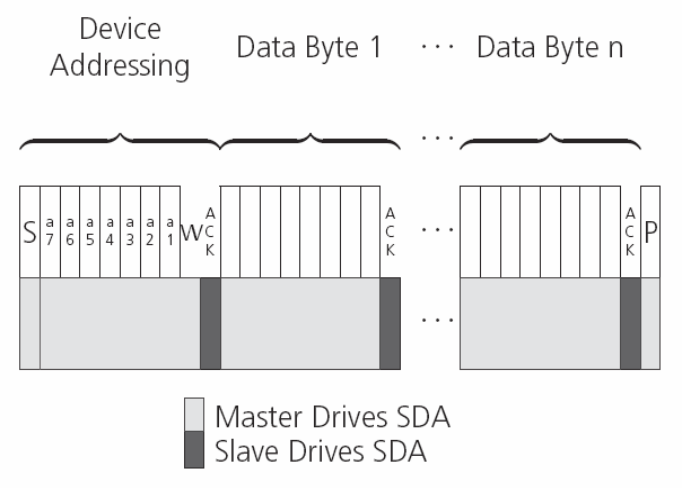


4.3.2 Write Access

The receiver does not provide any write access except for writing UBX messages (and NMEA messages) to the receiver, such as configuration or aiding data. Therefore, the register set mentioned in section Read Access is not writable. Following the start condition from the master, the 7-bit device address and the `RW` bit (which is a logic low for write access) are clocked onto the bus by the master transmitter. The receiver answers with an acknowledge (logic low) to indicate that it is responsible for the given address. Now, the master can write 2 to

N bytes to the receiver, generating a stop condition after the last byte being written. The number of data bytes must be at least 2 to properly distinguish from the write access to set the address counter in random read accesses.

DDC Write Access



4.4 SPI Port

A Serial Peripheral Interface (SPI) bus is available with selected receivers. See our online product selector matrix for availability.

SPI is a four-wire synchronous communication interface. In contrast to UART, the master provides the clock signal, which therefore doesn't need to be specified for the slave in advance. Moreover, a baud rate setting is not applicable for the slave. SPI modes 0-3 are implemented and can be configured using the field **mode.spiMode** in CFG-PRT for SPI (default is SPI mode 0).

The SPI clock speed is limited depending on hardware and firmware versions!

Maximum SPI clock speed

Generation	*Firmware*	*Max SPI speed*
u-blox 6	7	200 kHz
u-blox 6	6.02	100 kHz
u-blox 5	all	25 kHz

4.4.1 Read Access

As the register mode is not implemented for the SPI port, only the UBX/NMEA message stream is provided. This stream is accessed using the Back-To-Back Read and Write Access (see section Back-To-Back Read and Write Access). When no data is available to be written to the receiver, **MOSI** should be held logic high, i.e. all bytes written to the receiver are set to 0xFF.

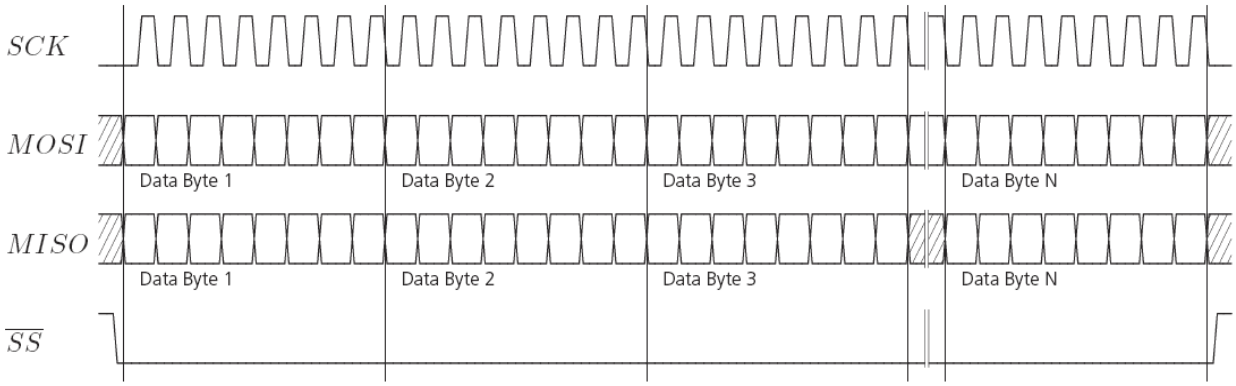
To prevent the receiver from being busy parsing incoming data, the parsing process is stopped after 50 subsequent bytes containing 0xFF. The parsing process is re-enabled with the first byte not equal to 0xFF. The number of bytes to wait for deactivation (50 by default) can be adjusted using the field **mode.ffCnt** in CFG-PRT for SPI, which is only necessary when messages shall be sent containing a large number of subsequent 0xFF bytes.

If the receiver has no more data to send, it sets **MISO** to logic high, i.e. all bytes transmitted decode to 0xFF. An efficient parser in the host will ignore all 0xFF bytes which are not part of a message and will resume data processing as soon as the first byte not equal to 0xFF is received.

4.4.2 Back-To-Back Read and Write Access

The receiver does not provide any write access except for writing UBX and NMEA messages to the receiver, such as configuration or aiding data. For every byte written to the receiver, a byte will simultaneous be read from the receiver. While the master writes to `MOSI`, at the same time it needs to read from `MISO`, as any pending data will be output by the receiver with this access. The data on `MISO` represents the results from a current address read, returning 0xFF when no more data is available.

SPI Back-To-Back Read/Write Access



4.5 How to change between protocols

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

- First of all, the preferred protocol(s) needs to be enabled on a port using `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 `CFG-PRT` messages.
- As a second step, activate certain messages on each port using `CFG-MSG`.

Despite the fact that concatenation of several configurations is still possible on receivers before u-blox 5, the use of this feature is discouraged as it won't work on receivers from u-blox 5 and above. u-blox 5 has 6 I/O ports, so backwards compatibility is dropped at this point.

5 Receiver Configuration

5.1 Configuration Concept

u-blox positioning technology is fully configurable with UBX protocol configuration messages (message class UBX-CFG). The configuration used by the GPS 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 receiver over an I/O port. The receiver will change its Current Configuration immediately after receiving the configuration message. The GPS receiver always uses only the Current Configuration.

Unless the Current Configuration is made permanent by using CFG-CFG as described below, the Current Configuration will be lost in case of (see message CFG-RST)

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

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).

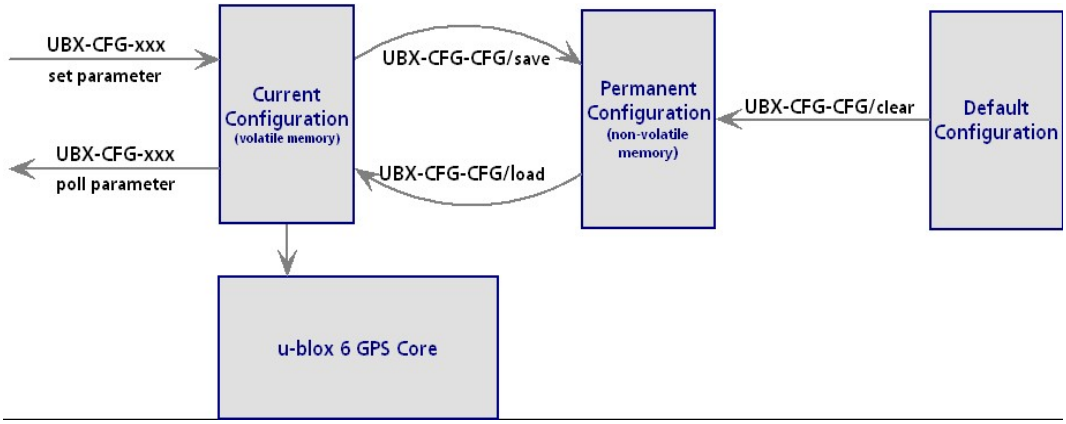
The Permanent Configurations are copied to the Current Configuration after start-up or when a UBX-CFG-CFG message with an appropriate **loadMask** (UBX-CFG-CFG/load) is sent to the receiver.

The Permanent Configuration can be restored to the receiver's Default Configuration by sending a UBX-CFG-CFG message with an appropriate **clearMask** (UBX-CFG-CFG/clear) to the receiver.

This only replaces the Permanent Configuration, not the Current Configuration. To make the receiver operate with the Default Configuration which was restored to the Permanent Configuration, a UBX-CFG-CFG/load command must be sent or the 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:



5.2 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.

Configuration sub-sections on u-blox 6

sub-section	*CFG messages*	*Description*
0	UBX-CFG-PRT UBX-CFG-USB	Port and USB settings
1	UBX-CFG-MSG	Message settings (enable/disable, update rate)
2	UBX-CFG-INF	Information output settings (Errors, Warnings, Notice, Test etc.)
3	UBX-CFG-NAV5 UBX-CFG-NAVX5 UBX-CFG-DAT UBX-CFG-RATE UBX-CFG-SBAS UBX-CFG-NMEA UBX-CFG-TMODE UBX-CFG-ESFGWT UBX-CFG-ESFDWT	Navigation Parameter, Receiver Datum, Measurement and Navigation Rate setting, Timemode settings, SBAS settings, NMEA protocol settings, ADR settings
4	UBX-CFG-TP UBX-CFG-TP2 UBX-CFG-RXM UBX-CFG-PM UBX-CFG-PM2	Power Mode Settings, Timepulse Settings
5-8	N/A	Reserved
9	UBX-CFG-RINV	Remote Inventory configuration
10	UBX-CFG-ANT	Antenna configuration
11-31	N/A	Reserved

5.3 Permanent Configuration Storage Media

The Current Configuration is stored in the receiver's volatile RAM. Hence, any changes made to the Current Configuration without saving will be lost in the events listed in the section above. 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 receiver.
- External FLASH memory, where available.
- External EEPROM (Electrically Erasable Programmable Read-Only Memory), where available via DDC (I2C compatible).
- External serial FLASH memory, where available via SPI.

When executing flash firmware, and writing configuration to flash device, the receiver will be stopped for the duration of the erase/write process. While this time communication through the interfaces will not be possible, and any input data might be lost.

5.4 Receiver Default Configuration

Permanent Configurations can be reset to Default Configurations through a UBX-CFG-CFG/clear message. The receiver's Default Configuration is determined at system startup. Refer to specific product data sheet for further details.

6 NMEA Protocol Configuration

The NMEA protocol on u-blox receivers can be configured to the need of customer applications using CFG-NMEA. By default all invalid positions out of the defined accuracy range are not reported.

There are two NMEA standards supported. The default NMEA protocol version is 2.3. Alternatively also Specification version 2.1 can be enabled (for details on how this affect the output refer to section Position Fix Flags in NMEA Mode).

NMEA filtering flags

Parameter	*Description*
Position filtering	If disabled, invalid or old position output is communicated, but the valid flag indicates that the data is not current.
Masked position filtering	If disabled, Masked position data is still output, but the valid flag will indicate that the defined accuracy range has been exceeded.
Time filtering	If disabled, the receiver's best knowledge of time is output, even though it might be wrong.
Date filtering	If disabled, the receiver's best knowledge of date is output, even though it might be wrong.
SBAS filtering	If enabled, SBAS satellites are not reported.
Track filtering	If disabled, unfiltered course over ground (COG) is output.

NMEA flags

Parameter	*Description*
Compatibility Mode	Some NMEA applications only work with a fixed number of digits behind the decimal point. Therefore u-blox receivers offer a compatibility mode to communicate with the most popular map 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.

7 Forcing a Receiver Reset

Typically, in GPS 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 this 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. Please note that some competitors 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 basically needs to download ephemeris until 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 exist. See the section on Aiding and Acquisition.
- **Hot start** In Hot start, 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.

The Reset Type can also be specified. This is not related to GPS, 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 pulling the Reset signal on the receiver.
- **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 GPS satellites.
- **Controlled Software Reset (GPS only)** only restarts the GPS tasks, without reinitializing the full system or reloading any stored configuration.
- **Controlled GPS Stop** stops all GPS tasks. The receiver will not be restarted, but will stop any GPS related processing.
- **Controlled GPS Start** starts all GPS tasks.

8 Remote Inventory

8.1 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 is possible to dump the data at startup.

8.2 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. More information about saving a configuration section can be found in chapter Configuration Concept.

9 Power Management

u-blox receivers support different power modes. These modes represent strategies of how to control the acquisition and tracking engines in order to achieve either the best possible performance or good performance with reduced power consumption.

Power modes are selected using the message CFG-RXM.

9.1 Maximum Performance Mode

During a Cold start, a receiver in Maximum Performance Mode continuously deploys the acquisition engine to search for all satellites. Once the receiver has a position fix (or if pre-positioning information is available), the acquisition engine continues to be used to search for all visible satellites that are not being tracked.

9.2 Eco Mode

During a Cold start, a receiver in Eco Mode works exactly as in Maximum Performance Mode. Once a position can be calculated and a sufficient number of satellites are being tracked, the acquisition engine is powered off resulting in significant power savings. The tracking engine continuously tracks acquired satellites and acquires other available or emerging satellites.

Note that even if the acquisition engine is powered off, satellites continue to be acquired.

9.3 Power Save Mode

Power Save Mode (PSM) allows a reduction in system power consumption by selectively switching parts of the receiver on and off. For possible restrictions concerning the power safe mode see Restrictions.

9.3.1 Operation

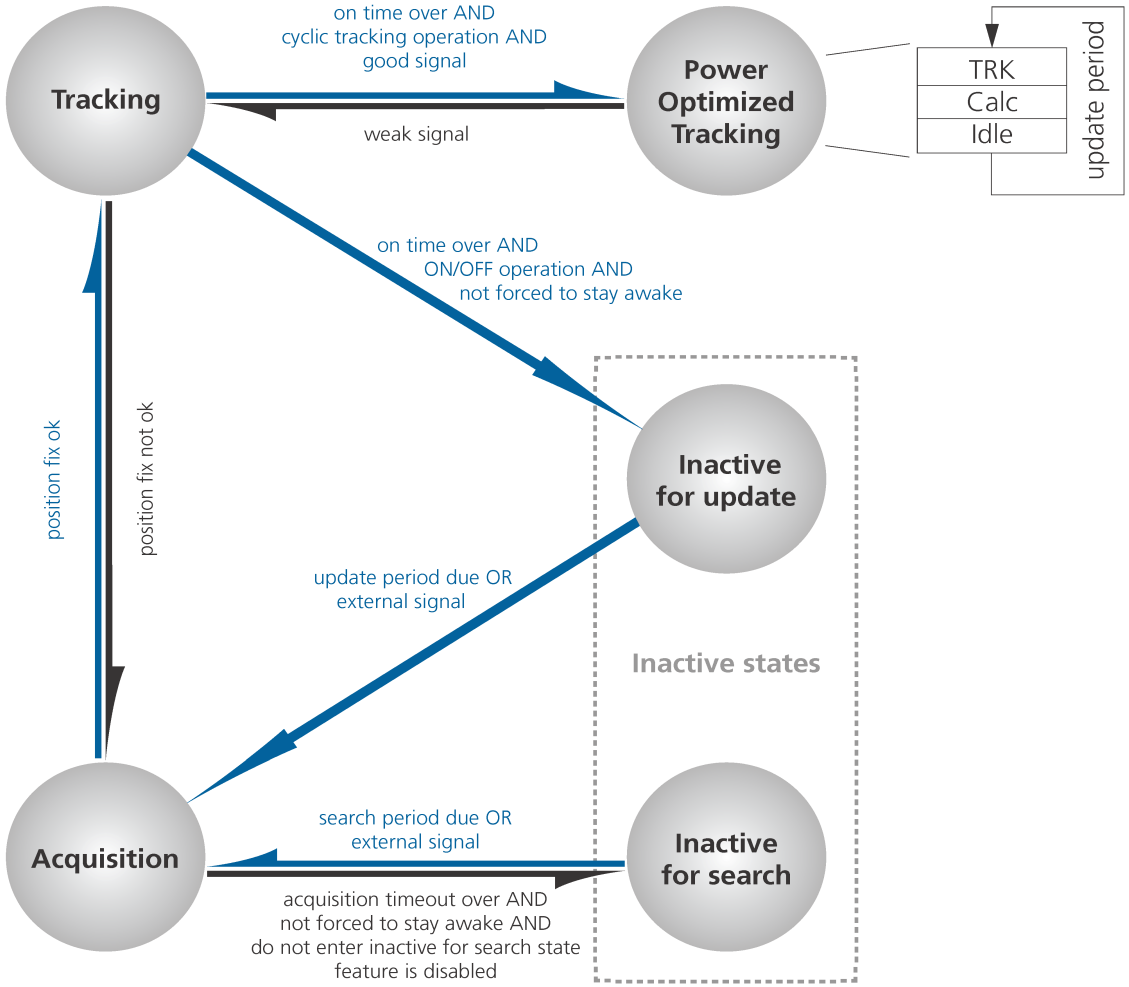
Power Save Mode has two modes of operation: cyclic tracking and ON/OFF operation. The mode of operation can be configured directly and depending on the setting, the receiver demonstrates different behavior. In cyclic tracking the receiver does not shut down completely between fixes, but uses low power tracking instead. Cyclic tracking operation is therefore used for short update periods. In ON/OFF operation the receiver switches between normal operation and a state of low or no activity. Hence, this mode of operation is suitable for long update periods.

PSM is based on a state machine with five different states: *Inactive for update* and *Inactive for search* states, *Acquisition* state, *Tracking* state and *Power Optimized Tracking (POT)* state.

- *Inactive* states: Most parts of the receiver are switched off.
- *Acquisition* state: The receiver actively searches for and acquires signals. Maximum power consumption.
- *Tracking* state: The receiver continuously tracks and downloads data. Less power consumption than in *Acquisition* state.
- *POT* state: The receiver repeatedly loops through a sequence of tracking (TRK), calculating the position fix (Calc), and entering an idle period (Idle). No new signals are acquired and no data is downloaded. Much less power consumption than in *Tracking* state.

The following figure illustrates the state machine:

State machine

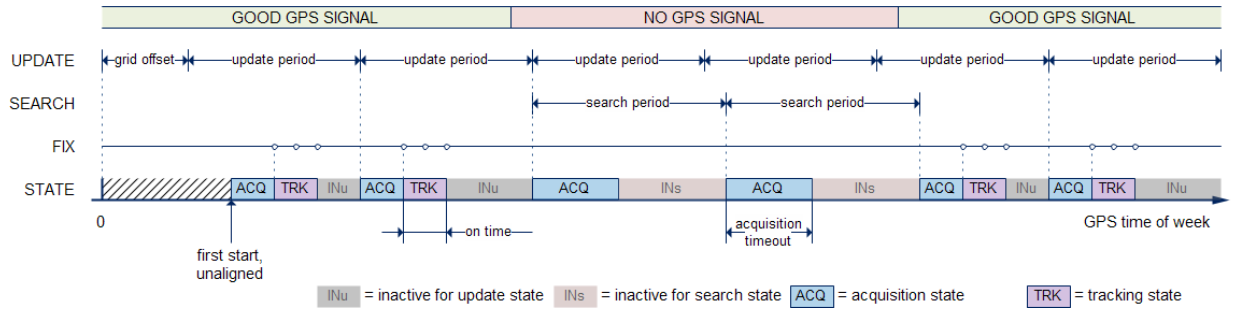


9.3.1.1 ON/OFF operation - long update period

When the receiver is switched on, it first enters *Acquisition* state. If it is able to obtain a position fix within the time given by the acquisition timeout, it switches to *Tracking* state. Otherwise it enters *Inactive for search* state and re-starts within the configured search grid. As soon as the receiver gets a position fix which is not masked (the masks, for example *3D only*, can be set using CFG-NAV5), it enters *Tracking* state. Upon entering *Tracking* state, the on time is started. Once the on time is over *Inactive for update* state is entered and the receiver re-starts according to the configured update grid. If the signal is lost while in *Tracking* state, *Acquisition* state is entered. If the signal is not found within the acquisition timeout, the receiver enters *Inactive for search* state. Otherwise the receiver will re-enter *Tracking* state and stay there until the newly started on time is over.

The diagram below illustrates how ON/OFF operation works:

Diagram of ON/OFF operation

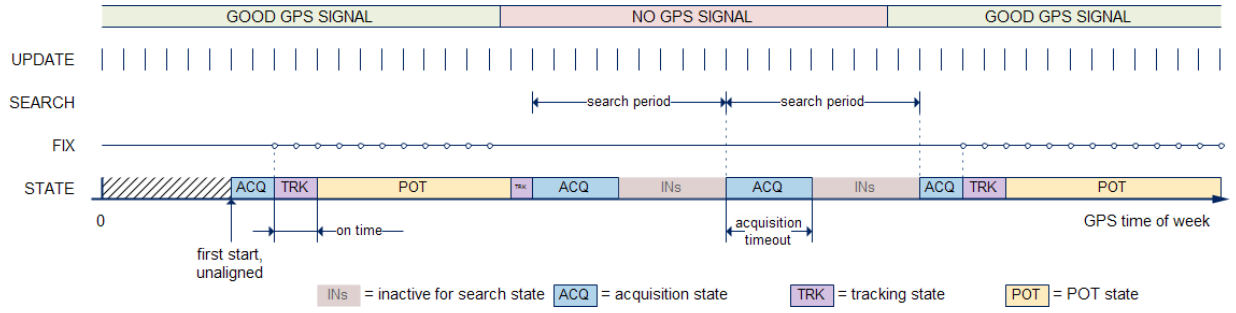


9.3.1.2 Cyclic tracking operation - short update period

When the receiver is switched on, it first enters *Acquisition* state. If it is able to obtain a position fix within the time given by the acquisition timeout, it switches to *Tracking* state. Otherwise, it will enter *Inactive for search* state and re-start within the configured search grid. After a valid position fix, *Tracking* state is entered and the on time is started. In other words the on time is started with the first position fix that is not masked (the masks, for example *3d only*, can be set with CFG-NAV5). Once the on time is over, *POT* state is entered. In *POT* state the receiver continues to output position fixes according to the update period. To have maximum power savings, set the on time to zero. This causes the receiver to enter *POT* state as soon as possible. If the signal becomes weak or is lost during *POT* state, *Tracking* state is entered. Once the signal is good again and the newly started on time is over, the receiver will re-enter *POT* state. If the receiver can't get a position fix in the *Tracking* state, it enters *Acquisition* state. Should the acquisition fail as well, *Inactive for search* state is entered.

The diagram below illustrates how cyclic tracking operation works:

Diagram of cyclic tracking operation



9.3.1.3 User controlled operation - update and search period of zero

Setting the update period to zero causes the receiver to wait in the *Inactive for update* state until woken up by the user. Setting the search period to zero causes the receiver to wait in the *Inactive for search* state indefinitely after an unsuccessful start-up. Any wake-up event will re-start the receiver. See chapter Wake-up for more information on wake-up events.

External wake-up is required when setting update or search period to zero!

9.3.1.4 Satellite data download

The receiver is not able to download satellite data (e.g. the ephemeris) while it is working in ON/OFF or cyclic tracking operation. Therefore it has to temporarily switch to continuous operation for the time the satellites transmit the desired data. To save power the receiver schedules the downloads according to an internal timetable and only switches to continuous operation while data of interest is being transmitted by the SVs.

Each SV transmits its own ephemeris data. Ephemeris data download is feasible when the corresponding SV

has been tracked with a minimal C/No over a certain period of time. The download is scheduled in a 30 minute grid or immediately when fewer than a certain number of visible SVs have valid ephemeris data.

Almanac, ionosphere, UTC correction and SV health data are transmitted by all SVs simultaneously. Therefore these parameters can be downloaded when a single SV is tracked with a high enough C/No.

9.3.2 Configuration

Power Save Mode is enabled and disabled with the UBX-CFG-RXM message and configured with the UBX-CFG-PM2 message.

When enabling Power Save Mode, SBAS support can be disabled (UBX-CFG-SBAS) since the receiver will be unable to download any SBAS data in this mode.

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

Power Save Mode configuration options

Parameter	*Description*
Mode of operation	Receiver mode of operation
Update period	Time between two position fix attempts
Search period	Time between two acquisition attempts if the receiver is unable to get a position fix
Acquisition timeout	Time after which the receiver stops acquisition and enters *Inactive for search* state
On-time	Time the receiver remains in *Tracking* state and produces position fixes
Wait for timefix	Wait for time fix before entering *Tracking* state
Do not enter *Inactive for search* state	Receiver does not enter *Inactive for search* state if it can't get a position fix but keeps trying instead
Update RTC	Enables periodic Real Time Clock (RTC) update
Update Ephemeris	Enables periodic ephemeris update
EXTINT selection	Selects EXTINT pin used with pin control feature
EXTINT 'high' keeps awake	Enables force-ON pin control feature
EXTINT 'low' forces sleep	Enables force-OFF pin control feature
Grid offset	Time offset of update grid with respect to GPS start of week

9.3.2.1 Mode of operation

The mode of operation to use mainly depends on the update period: For short update periods (in the range of a few seconds), cyclic tracking should be configured. On the other hand, for long update periods (in the range of minutes or longer) only work with ON/OFF operation.

See chapter ON/OFF operation - long update period and Cyclic tracking operation - short update period for more information on the two modes of operation.

9.3.2.2 Update and search period

The update period specifies the time between successive position fixes. If no position fix can be obtained within the acquisition timeout, the receiver will retry after the time specified by the search period. Update and search period are fixed with respect to an absolute time grid based on GPS time. They do not refer to the time of the last valid position fix or last position fix attempt. For possible restrictions see Restrictions.

New settings are ignored if the update period or the search period exceeds the maximum number of milliseconds in a week. In that case the previously stored values remain effective.

9.3.2.3 Acquisition timeout

The receiver tries to obtain a position fix within the time given in the acquisition timeout. This setting is treated as a minimum value. If the receiver determines that it needs more time for the given starting conditions, it will automatically prolong this time. If set to zero, the acquisition timeout is exclusively determined by the receiver. In case of a very weak or no GPS signal, the timeout determined by the receiver may be shortened in order to save power. However, the acquisition timeout will never be shorter than the configured value.

9.3.2.4 On time and wait for timefix

The on time specifies how long the receiver stays in *Tracking* state before switching to *POT* and *Inactive for update* state respectively. The quality of the position fixes can be configured by setting the masks in the message UBX-CFG-NAV5. If the wait for timefix option is enabled the transition from *Acquisition* to *Tracking* state is made only if the time is known and within the configured limits, and the receiver is continuously producing position fixes for more than two seconds. Thus enabling the wait for timefix option usually delays the transition from *Acquisition* to *Tracking* state by a few seconds. Keep in mind that setting harder limits in UBX-CFG-NAVX5 will prolong start-up time so you might want to increase the acquisition timeout.

9.3.2.5 Do not enter 'inactive for search' state when no fix

If this option is enabled, the receiver acts differently in case it can't get a fix: instead of entering *Inactive for search* state, it keeps trying to acquire a fix. In other words, the receiver will never be in *Inactive for search* state and therefore the search period and the acquisition timeout are obsolete.

9.3.2.6 Update RTC and Eph

To maintain the ability of a fast start-up, the receiver needs to calibrate its RTC and update its ephemeris data on a regular basis. This can be ensured by activating the update RTC and update Eph option. The RTC is calibrated every 5 minutes and the ephemeris data is updated approximately every 30 minutes. See chapter Satellite data download for more information.

9.3.2.7 EXTINT pin control

The pin control feature allows overriding the automatic active/inactive cycle of Power Save Mode. The state of the receiver can be controlled through either the EXTINT0 or the EXTINT1 pin.

If the Force-ON feature is enabled, the receiver will not enter the *Inactive* states as long as the configured EXTINT pin (either EXTINT0 or EXTINT1) is at a 'high' level. The receiver will therefore always be in *Acquisition/Tracking* states (ON/OFF operation) and *Acquisition/Tracking/POT* states (cyclic tracking operation) respectively. When the pin level changes to 'low' the receiver continues with its configured behavior. UBX-CFG-PM2 is used to select and configure the pin that will control the behavior as described above.

If the Force-OFF feature is enabled, the receiver will enter *Inactive* state (with a delay of up to five seconds) and remain there until the next wake-up event. Any wake-up event can wake up the receiver, even while the EXTINT pin is set to Force-OFF. However, the receiver will only wake up for the time period needed to read the configuration pin settings, i.e. Force-OFF, and will then enter *Inactive* state again.

9.3.2.8 Grid offset

Once the receiver has a valid time, the update grid is aligned to the start of the GPS week (Sunday at 00:00 o'clock). Before having a valid time, the update grid is unaligned. A grid offset now shifts the update grid with respect to the start of the GPS week. An example of usage can be found in chapter Use grid offset.

The grid offset does not work in cyclic tracking operation.

9.3.2.9 Restrictions

The following restrictions apply to firmware version 7.03 only. On firmware 7.03, on/off operation is not available. Further, there are some restriction on the possible update periods when using cyclic operation:

- For all TCXO based receivers/designs the cyclic tracking update period can be 1..10s without restrictions.
- For all crystal based u-blox GPS modules, the cyclic tracking update period should be 1..3s.
- For crystal based chip designs, a cyclic tracking update period of 1s is recommended.

In general, an update period of 1s is recommended.

9.3.3 Communication, wake-up, FixNow interface, USB and AssistNow Autonomous

9.3.3.1 Communication

When PSM is enabled, communication with the receiver (e.g. UBX message to disable PSM) requires particular attention. This is because the receiver may be in *Inactive* state and therefore unable to receive any message through its interfaces. To ensure that the configuration messages are processed by the receiver, even while in *Inactive* 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 the receiver up in case it is in *Inactive* state. If the receiver is not in *Inactive* 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. On the other hand, if the interval is too long, the receiver may return to *Inactive* state before the configuration message was received.
- Send the configuration save message immediately after the configuration message.

9.3.3.2 Wake-up

The receiver can be woken up 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 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.

9.3.3.3 FixNow interface

The CFG-FXN message is still accepted, *but may be discontinued in future versions of the software.*

Do not use UBX-CFG-FXN message for new designs.

Since u-blox 5/6 Power Management has other configuration parameters than FixNow, the parameters of CFG-FXN message have to be mapped to those of the CFG-PM2 message. The following tables show how the mapping is done.

FXN to PM parameter mapping with "FXN On/Off Time" enabled

Power Management parameter	*FixNow parameter(s)*	*Default Value*
Update Period	T_on + T_off	-
On-time	T_on	-
Search Period	T_acq + T_acq_off	-
Min acq.time	T_acq	-
Grid Offset	Base TOW	-

FXN to PM parameter mapping with "FXN On/Off Time" enabled continued

Power Management parameter	*FixNow parameter(s)*	*Default Value*
Wait for Timefix	-	Disabled
Update RTC	-	Disabled
Update Ephemeris	-	Disabled
EXTINT Selection	-	EXTINT0
EXTINT Forces ON	-	Disabled
EXTINT Forces OFF	-	Disabled

For possible restrictions (Update Period) see Restrictions.

FXN to PM parameter mapping with "FXN On/Off Time" disabled

Power Management parameter	*FixNow parameter(s)*	*Default Value*
Update Period	-	1000 [ms]
On-time	T_on	-
Search Period	T_acq + T_acq_off	-
Min acq.time	T_acq	-
Grid Offset	-	0
Wait for Timefix	-	Disabled
Update RTC	-	Disabled
Update Ephemeris	-	Disabled
EXTINT Selection	-	EXTINT0
EXTINT Forces ON	-	Disabled
EXTINT Forces OFF	-	Disabled

System mode is always set to backup. If use on/off time is not enabled, update period is set to 1 s. This causes the receiver to operate in cyclic tracking.

9.3.3.4 behavior while USB host connected

As long as the receiver is connected to a USB host, it will not enter backup state. Instead, CPU-on state is entered. This assures that the USB specification is not violated. The drawback, however, is that power consumption is higher.

Wake-up by pin/UART/USB is possible even if the receiver is connected to a USB host. The state of the pin must be changed for at least one millisecond.

9.3.3.5 Cooperation with the AssistNow Autonomous feature

If both PSM and AssistNow Autonomous features are enabled, the receiver won't enter *Inactive for update* state as long as *AssistNow Autonomous* carries out calculations. This prevents losing data from unfinished calculations and, in the end, reduces the total extra power needed for *AssistNow Autonomous*. The delay before entering *Inactive for update* state, if any, will be in the range of several seconds, rarely more than 20 seconds.

Only entering *Inactive for update* state is affected by *AssistNow Autonomous*. In other words: in cyclic tracking operation, *AssistNow Autonomous* will not interfere with the PSM (apart from the increased power consumption).

Enabling the AssistNow Autonomous feature will lead to increased power consumption while prediction is calculated. The main goal of PSM is to reduce the overall power consumption. Therefore for each application special care must be taken to judge whether AssistNow Autonomous is beneficial to the overall power consumption or not.

9.3.4 Examples

9.3.4.1 Use Grid Offset

Scenario: Get a position fix once a day at a fixed time. If the position fix cannot be obtained try again every two hours.

Solution: First set the update period to 24*3600s and the search period to 2*3600s. Now a position fix is obtained every 24 hours and if the position fix fails retrials are scheduled in two hour intervals. As the update grid is aligned to midnight Saturday/Sunday, the position fixes happen at midnight. By setting the grid offset to 12*3600s the position fixes are shifted to once a day at noon. If the position fix at noon fails, retrials take place every two hours, the first at 14:00. Upon successfully acquiring a position fix the next fix attempt is scheduled for noon the following day.

9.3.4.2 Use update periods of zero

Scenario: Get a position fix on request.

Solution: Set update and search period to zero. This way the receiver stays inactive until it is woken up.

9.4 Peak current settings

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

This setting is independent of the activated mode (Maximum Performance, Eco or Power Save Mode).

9.5 Power On/Off command

With message RXM-PMREQ the receiver can be forced to enter backup state. It will stay in backup state for the time specified in the message or until it is woken up by an EXTINT or activity on the RXD1 line.

Sending the message RXM-PMREQ while the receiver is in Power Save Mode will overrule PSM and force the receiver to enter Inactive state. It will stay in Inactive state until woken up. After wake-up the receiver continues working in Power Save Mode as configured.

10 Time Mode Configuration

This section relates to the configuration messages CFG-TMODE and CFG-TMODE2.

10.1 Introduction

Time Mode is a special stationary GPS receiver mode where the position of the receiver is known and fixed and only the time is calculated using all available satellites. This mode allows for maximum time accuracy as well as for single-SV solutions.

10.2 Fixed Position

In order to use the *Time Mode*, the receiver's position must be known as exactly as possible. Either the user already knows and enters the position, or it is determined using a Survey-in. Errors in the fixed position will translate into time errors depending on the satellite constellation. Using the TDOP value (see **UBX-NAV-DOP**) and assuming a symmetrical 3D position error , the expected time error can be estimated as

```
time error = tdop * position error
```

As a rule of thumb the position should be known with an accuracy of better than 1 m for a timing accuracy in the order of nanoseconds. If an accuracy is required only in the order of microseconds, a position accuracy of roughly 300 m is sufficient.

10.3 Survey-in

Survey-in is the procedure of determining a stationary receiver's position prior to using *Time Mode* by averaging. The current implementation builds a weighted mean of all valid 3D position solutions. Two stop criteria can be specified:

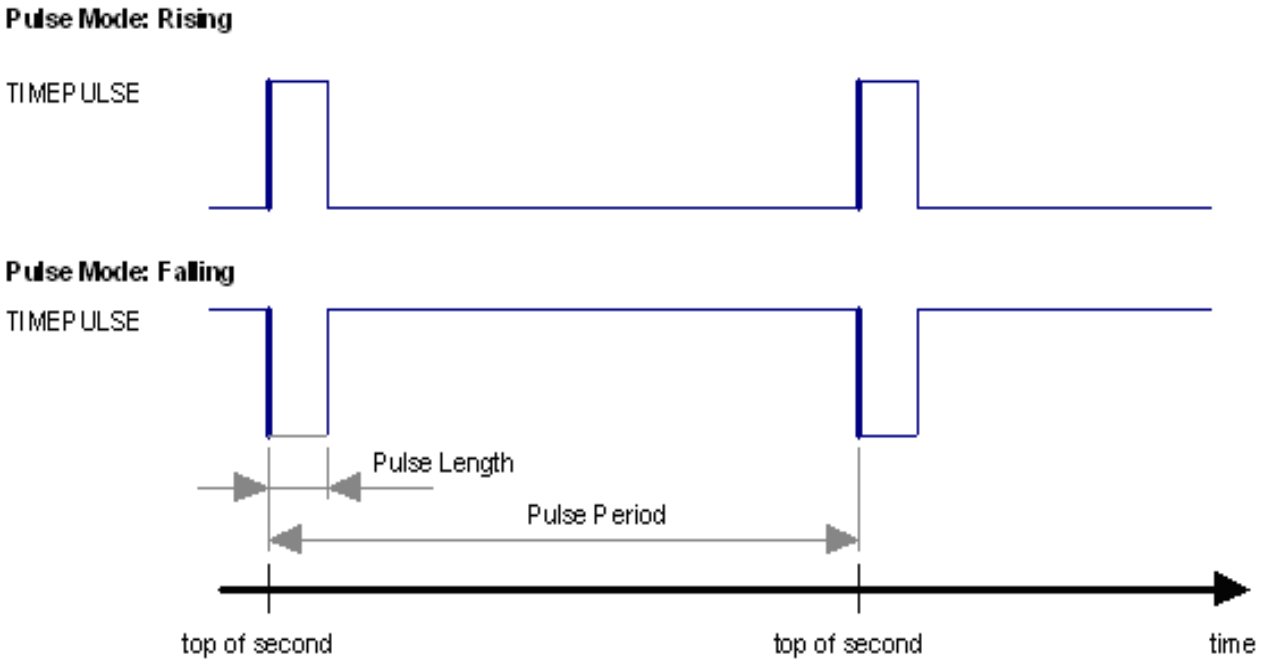
- The **minimum observation time** defines a minimum amount of observation time regardless of the actual number of valid fixes that were used for the position calculation. Reasonable values range from one day for high accuracy requirements to a few minutes for coarse position determination.
- The **required 3D position standard deviation** forces the calculated position to be of at least the given accuracy. As the position error translates into a time error when using *Time Mode* (see above), one should carefully evaluate the time accuracy requirements and the choose an appropriate position accuracy requirement.

Survey-in ends, when **both** requirements are met. After Survey-in has finished successfully, the receiver will automatically enter fixed position *Time Mode*. The Survey-in status can queried using the `UBX-TIM-SVIN` message.

The "Standard Deviation" parameter defines uncertainty of the manually provided "True Position" set of parameters. This uncertainty directly affects the accuracy of the timepulse. This is to prevent an error that would otherwise be present in the timepulse because of the initially inaccurate position (assumed to be correct by the receiver) without users being aware of it. The "3D accuracy" parameter in "Fixed Position" as well as the "Position accuracy limit" in "Survey-in" affect the produced time information and the timepulse in the same way. Please note that the availability of the position accuracy does not mitigate the error in the timepulse but only accounts for it when calculating the resulting time accuracy.

11 Timepulse

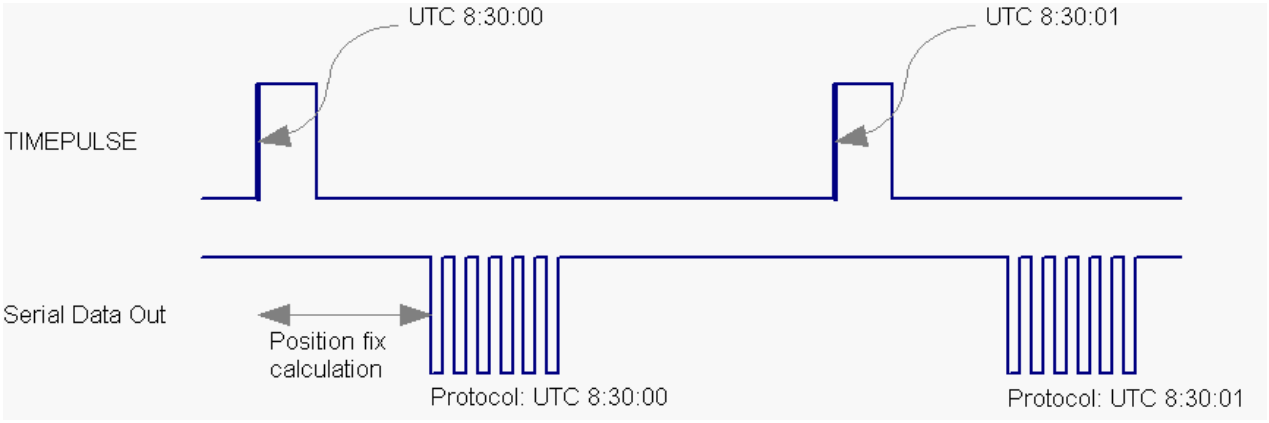
u-blox GPS receivers include a Timepulse function providing clock pulses with configurable duration and frequency. The UBX-TIM-TP message provides time information for the next pulse, time source and the quantization error of the output pin.



11.1 Recommendations

- For best timepulse performance it is recommended to disable the SBAS subsystem.
- When using Timepulse for Precision Timing applications it is recommended to calibrate the RF signal delay against a reference-timing source.
- Care needs to be given to the Cable Delay settings in the receiver configuration.
- In order to get the best timing accuracy with the antenna, a fixed and *accurate* position is needed. Once the receiver is in timing mode, the dynamic model does not influence the timing accuracy.
- If relative time accuracy between multiple receivers is required, do not mix receivers of different product families. If this is required, the receivers must be calibrated by accordingly setting cable delay and user delay.

The sequential order of the signal present at the TIMEPULSE pin and the respective output message for the simple case of 1 pulse per second (1PPS) and a one second navigation update rate is shown in the following figure.



11.2 Timepulse Configuration (u-blox 6)

u-blox 6 receivers provide one or two (e.g. LEA-6T) TIMEPULSE pins delivering a Timepulse (TP) signal with a configurable pulse period, pulse length and polarity (rising or falling edge). Check the product data sheet for detailed specification of configurable values.

It is possible to define different signal behavior (i.e. output frequency and pulse length) depending on whether or not the receiver is locked to GPS time. Timepulse signals can be configured using the UBX proprietary message CFG-TP5. In addition, the UBX message CFG-TP is also available to change settings. This message is provided for legacy purposes, and it is recommended to use CFG-TP5.

11.3 Configuring Timpulse with UBX-CFG-TP5

The UBX message CFG-TP5 can be used to change the Timepulse settings, and includes the following parameters defining the pulse:

- **timepulse index** - Index of Timepulse.
- **antenna cable delay** - Signal delay due to the cable between antenna and receiver.
- **RF group delay** - Signal delay in the RF module of the receiver (read-only).
- **pulse frequency/period** - Frequency or period time of the pulse.
- **pulse frequency/period lock** - Frequency or period time of the pulse, as soon as receiver has calculated a valid time from a received signal. Only used if the according flag is set to use another setting in locked mode.
- **pulse length/ratio** - Length or duty cycle of the generated pulse, either specifies a time or ratio for the pulse to be on/off.

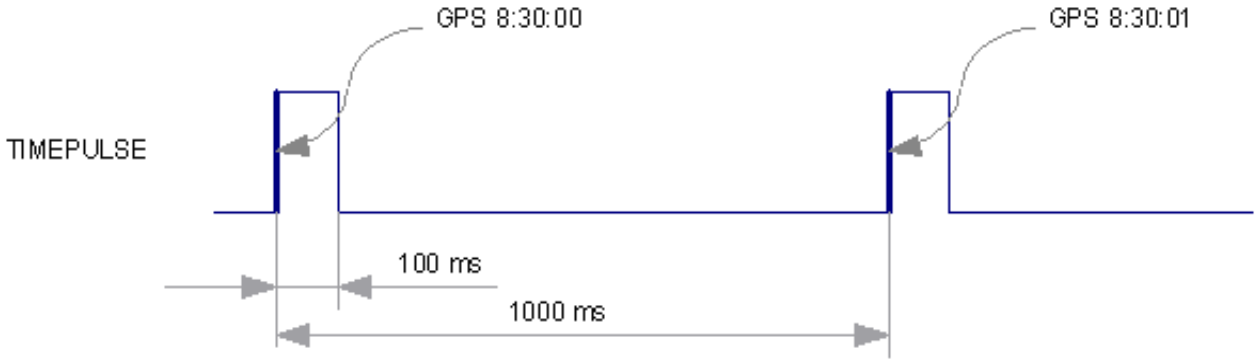
- **pulse length/ratio lock** - Length or duty cycle of the generated pulse, as soon as receiver has calculated a valid time from a received signal. Only used if the according flag is set to use another setting in locked mode.
- **user delay** - The cable delay from u-blox 6 receiver to the user device plus signal delay of any user application.
- **active** - Timepulse will be active if this bit is set.
- **lock to gps freq** - Use frequency gained from GPS signal information rather than local oscillator's frequency if flag is set.
- **locked other setting** - If this bit is set, as soon as the receiver can calculate a valid time, the alternative setting is used. This mode can be used for example to disable Timepulse if time is not locked, or indicate lock with different duty cycles.
- **is frequency** - Interpret the 'Frequency/Period' field as frequency rather than period if flag is set.
- **is length** - Interpret the 'Length/Ratio' field as length rather than ratio if flag is set.
- **align to TOW** - If this bit is set, pulses are aligned to the top of a second. Alignment is only possible with an integer count of pulses fitting into one second, if it does not and the bit is set, it will be cleared by the receiver.
- **polarity** - If set, the first edge of the pulse is a rising edge.
- **grid UTC/GPS** - Selection between UTC (0) or GPS (1) timegrid. Also effects the time output by TIM-TP message.

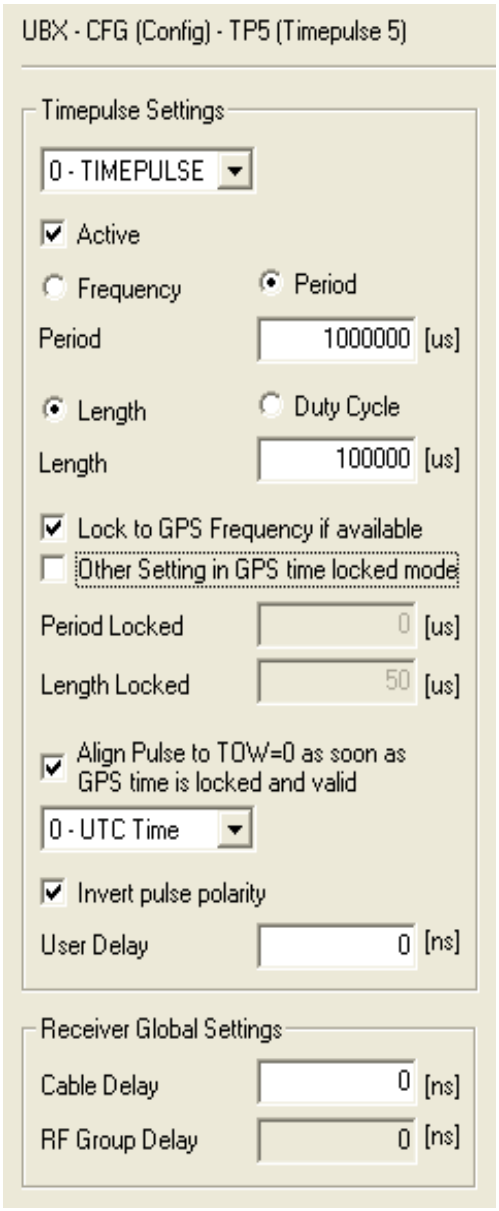
The pulse interval must be an integer fraction of 1 second to enable alignment to TOW. The maximum pulse length can't exceed the pulse period.

Timepulse settings shall be chosen in such a way, that neither the high nor the low period of the output is less than 50 ns (except when disabling it completely), otherwise pulses can be lost.

11.3.1 Example 1:

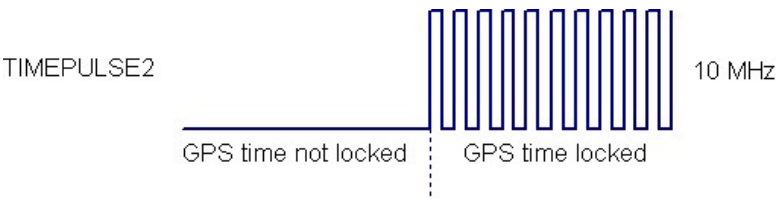
The example below shows the 1PPS TP signal generated on the TIMEPULSE output according to the specific parameters of the CFG-TP5 message. The 1 Hz output is maintained whether or not the receiver is locked to GPS time. The alignment to TOW can only be maintained when GPS time is locked.

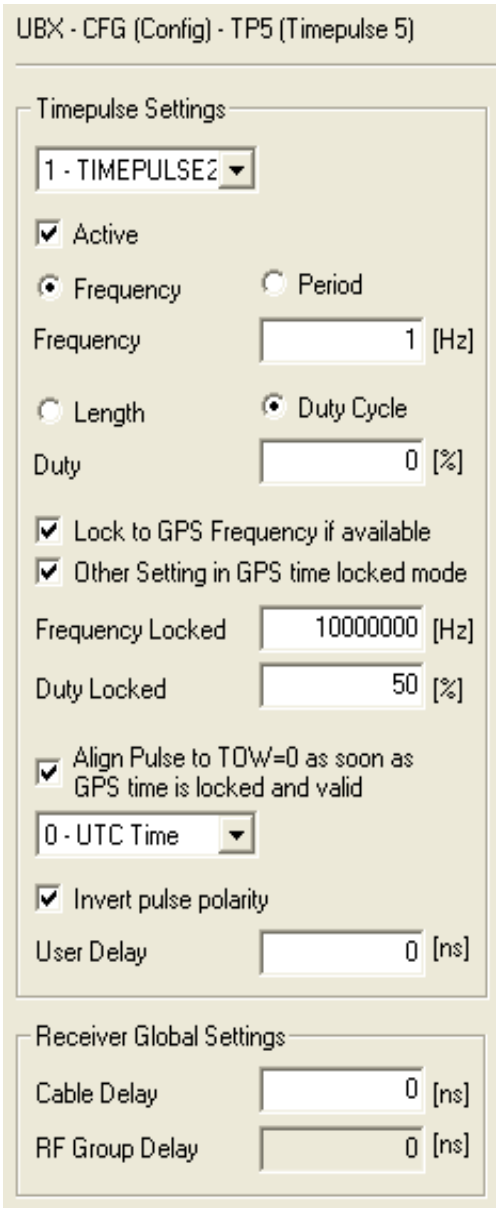




11.3.2 Example 2:

The following example shows a 10 MHz TP signal generated on the TIMEPULSE2 output when the receiver is locked to GPS time. Without the lock to GPS time no frequency is output.





11.4 Configuring Timpulse with UBX-CFG-TP

The CFG-TP message comprises the following parameters defining the hardware-synchronized Timepulse signal:

- **pulse interval** - time interval between pulses
- **pulse length** - duration of the pulse (time period between rising and falling edge)
- **pulse mode** - if not disabled the pulse synchronization can be configured to be done on rising or falling edge
- **time reference** - the reference time source (time base) used for pulse synchronization and pulse time given in TIM-TP output message
- **synchronization mode** - the pulse can be configured to be always synchronized and will be available only in this case. If the pulse is allowed to be asynchronized it will be available at any time even when the time is not valid
- **antenna cable delay** - the signal delay due to the cable between antenna and receiver

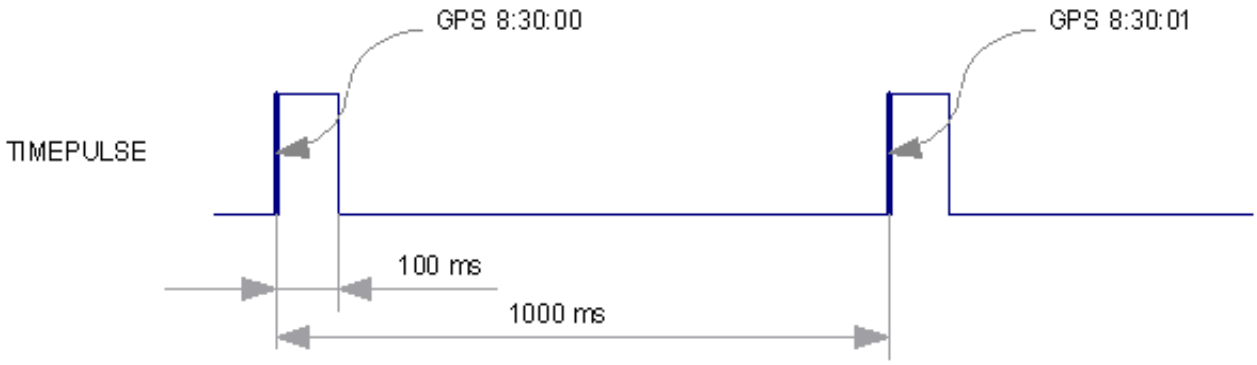
- **RF group delay** - delay of the signal in the RF module of the u-blox 5 receiver (hard coded)
- **user delay** - the cable delay from u-blox 5 receiver to the user device plus signal delay of any user application

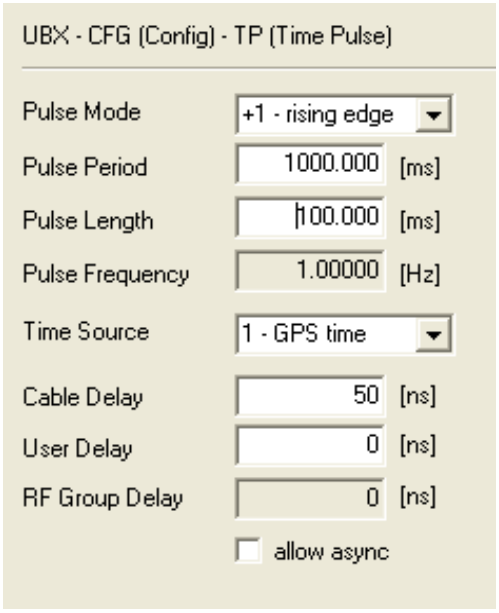
The pulse interval must be an integer division of 60 seconds. This represents the time of the receiver's re-synchronization, and if the declared number of pulses within this period is not a whole number, a stream of non-equidistant timepulses is generated.

The maximum pulse length shall not exceed the pulse period minus 1 microsecond. Timepulse is only output when the receiver has determined the time with sufficient accuracy and reliability.

11.4.1 Example:

The example shows the 1PPS TP signal generated according the specific parameters of the CFG-TP message.





12 Receiver Status Monitoring

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

The main purposes are

- Stack and CPU load (Antaris 4 only)
- Hardware and Software Versions, using MON-VER
- Status of the Communications Input/Output system

- Status of various Hardware Sections with MON-HW

12.1 Input/Output system

The I/O system is a GPS-internal layer where all data input- and output capabilities (such as UART, DDC, SPI, USB) of the GPS 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 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 GPS receiver is placed in buffers. Usage of these buffers is shown with the message MON-RXBUF. Further, as data is then decoded within the receiver (e.g. to separate UBX and NMEA data), the 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.

A `target` in the context of the I/O system is a I/O port. The following table shows the target numbers used:

Target Number assignment

Target #	*Electrical Interface*
0	DDC (I2C compatible)
1	UART 1
2	UART 2
3	USB
4	SPI
5	reserved

Protocol Number assignment

Protocol #	*Protocol Name*
0	UBX Protocol
1	NMEA Protocol
2	RTCM Protocol (not supported on u-blox 5)
3	RAW Protocol (not supported on u-blox 5)
4..7	Reserved for future use

12.2 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 the application and then calibrate the 'not jammed' case. If the value rises significantly above this threshold, this indicates that a continuous wave jammer is present.

This indicator is always enabled.

12.3 Jamming/Interference Monitor

The field `jammingState` of the 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	jammer monitor not enabled, uninitialized or antenna disconnected
1	OK	no interference detected
2	Warning	position ok but interference is visible (above the thresholds)
3	Critical	no reliable position fix with interference 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.

Jamming/Interference monitor is only supported by u-blox 6 Firmware 7.01 and above. This functionality is not supported in Power Save Mode (PSM).

13 Aiding and Acquisition

13.1 Introduction

The UBX Message Class AID provides all mechanisms for providing Assisted GPS Data to u-blox GPS receivers, including AssistNow Online and AssistNow Offline.

13.2 Startup Strategies

- **Cold start:** In this startup mode, the receiver has no information about last position, time, velocity, frequency etc. Therefore, the receiver has to search the full time- and frequency space, and also all possible satellite numbers. If a satellite signal is found, it is being tracked to decode ephemeris (18-36 seconds under strong signal conditions), whereas the other channels continue to search satellites. Once there are sufficient number of satellites with valid ephemeris, the receiver can calculate position- and velocity data. Note that some competitors call this startup mode Factory Startup.
- **Warm start:** In Warm start mode, the receiver has approximate information of time, position, and coarse data on Satellite positions (Almanac). In this mode, after power-up, the receiver basically needs to download ephemeris until it can calculate position- and velocity data. As the ephemeris data usually is outdated after 4 hours, the receiver will typically start with a warmstart if it was powered down for more than that amount of time. For this scenario, several augmentations exist. See the sections on AssistNOW online and offline below.
- **Hot start:** In Hot start, 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 that, the CFG-RST message offers the

navBbrMaskfield, where Hot, Warm and Cold starts can be initiated, and also other combinations thereof.

13.3 Aiding / Assisted GPS (A-GPS)

The Challenge of Stand-alone GPS

GPS users expect instant position information. With standard GPS this is not always possible because at least four satellites must transmit their precise orbital position data, called Ephemeris, to the GPS receiver. Under adverse signal conditions, data downloads from the satellites to the receiver can take minutes, hours or even fail altogether.

Assisted GPS (A-GPS) boosts acquisition performance by providing data such as Ephemeris, Almanac, accurate time and satellite status to the GPS receiver via mobile networks or the Internet. The aiding data enables the receiver to compute a position within seconds, even under poor signal conditions.

13.4 Aiding Data

The following aiding data can be submitted to the receiver:

- **Position:** Position information can be submitted to the receiver using the UBX-AID-INI message. Both, ECEF X/Y/Z and latitude/longitude/height formats are supported.
- **Time:** The 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. Both methods are supported in the UBX-AID-INI 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-AID-INI message.
- **Orbit data:** Orbit data can be submitted using UBX-AID-ALM and UBX-AID-EPH.
- **Additional information:** UBX-AID-HUI can be used to supply health information, UTC parameters and ionospheric data to the receiver.

13.5 Aiding Sequence

A typical aiding sequence comprises the following steps:

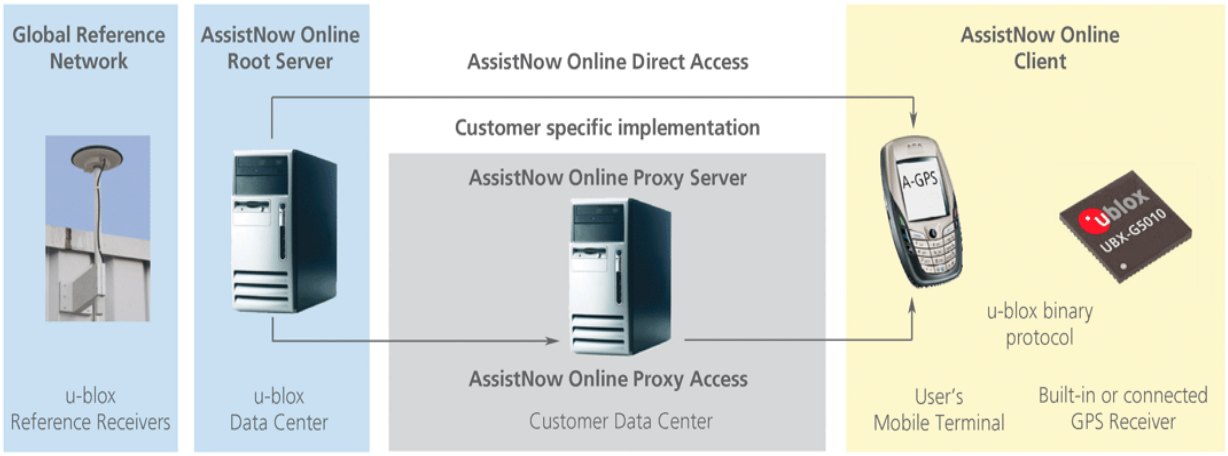
- Power-up the GPS receiver
- Send UBX-AID-INI (time, clock and position) message.
- Send UBX-AID-EPH (ephemeris) message.
- Apply optional hardware time synchronization pulse within 0.5 s after (or before, depending on the configuration in UBX-AID-INI) sending the UBX-AID-INI message if hardware time synchronization is required. When sending the message before applying the pulse, make sure to allow the GPS receiver to parse and process the aiding message. The time for parsing depends on the baud rate. The processing time is 100 ms maximum.
- Send optional UBX-AID-HUI (health, UTC and ionosphere parameters) message.
- Send optional UBX-AID-ALM (almanac) message.

13.6 AssistNow Online

AssistNow Online is u-blox' end-to-end Assisted GPS (A-GPS) solution that boosts GPS acquisition performance, bringing Time To First Fix (TTFF) down to seconds. The system works by accessing assistance data such as Ephemeris, Almanac and accurate time from our Global Reference Network of GPS receivers placed around the

globe. With A-GPS, the receiver can acquire satellites and provide accurate position data instantly on demand, even under poor signal conditions.

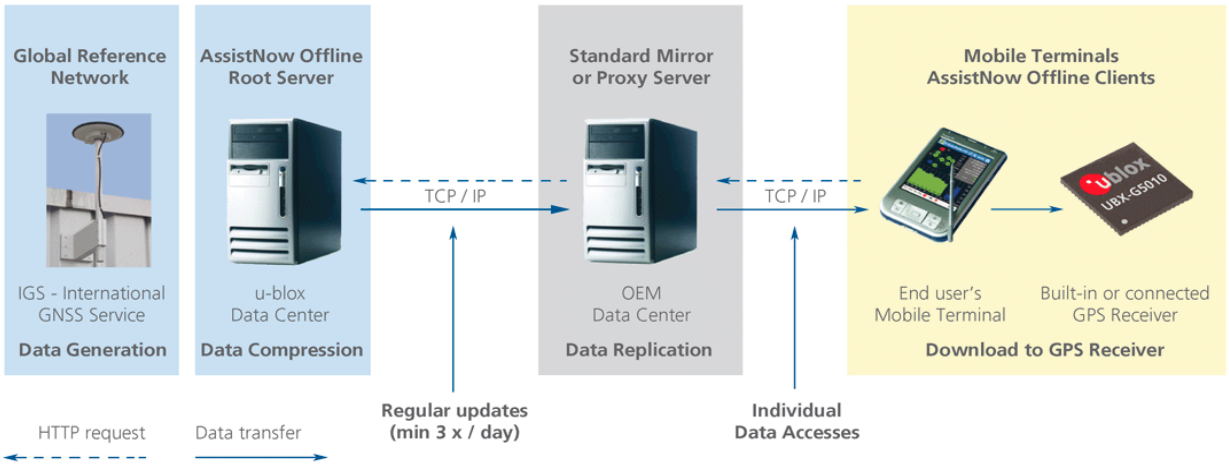
AssistNow Online makes use of User Plane communication and open standards such as TCP/IP. Therefore, it works on all standard mobile communication networks that support Internet access, including GPRS, UMTS and Wireless LAN. No special arrangements need to be made with mobile network operators to enable AssistNow Online.



In terms of the messages AssistNow Online consists of Aiding data which deliver Position and Time `UBX-AID-INI`, Ephemerides `UBX-AID-EPH`, Almanac `UBX-AID-ALM` and Health/UTC/Iono information `UBX-AID-HUI`

13.7 AssistNow Offline

AssistNow Offline is an A-GPS service that boosts GPS acquisition performance, bringing Time To First Fix (TTFF) down to seconds. Unlike AssistNow Online, this solution enables instant positioning without the need for connectivity at start-up. The system works by using AlmanacPlus (ALP) differential almanac correction data to speed up acquisition, enabling a position fix within seconds. Users access the data by means of occasional Internet downloads, at the user's convenience.



u-blox provides AlmanacPlus data files in different sizes, which contain differential almanac corrections that are valid for a period of between 1 and 14 days thereafter. Users can download correction data anytime they have an Internet connection. The GPS receiver stores the downloaded data in the non-volatile memory. As an alternative, a host CPU may store the file, but deliver the data in pieces when requested.

AssistNow Offline works in locations without any wireless connectivity as the correction data files reside in the receiver or the host. This makes them immediately available upon start-up, eliminating connection set-up delays, download waiting times and call charges.

The simplest set-up is for GPS receivers including an internal Flash Memory or an external SPI Flash Memory where ALP data can be stored. In this case, the UBX-AID-ALP message is used.

When the GPS receiver has neither an internal Flash Memory nor an external SPI Flash Memory, the ALP file must be stored to the host CPU. The GPS receiver can then request data from the host when needed. This arrangement is implemented using the UBX-AID-ALPSRV message.

In both cases, status reporting on ALP data currently available to the GPS receiver can be taken from message UBX-AID-ALP (STAT).

AssistNow Offline data are published at http://alp.u-blox.com/.

13.7.1 Flash-based AlmanacPlus Overview

Flash-based AlmanacPlus functionality means that AlmanacPlus data is stored in the program flash memory connected to the u-blox 6 chip.

The task of a server is simply to download the data from an Internet server or other sources, and then deliver the full file piece by piece to the GPS receiver. This is different to the method described in UBX-AID-ALPSRV where the file would remain within the host and the GPS receiver would request chunks from that file when needed.

The message AID-ALP exists in several variants, combining all functionality needed to download data and report status within one Class/Message ID.

13.7.1.1 Download Procedure

The following steps are a typical sequence for downloading an ALP file to the receiver:

- The server downloads a copy of a current ALP file, and stores it locally
- It sends the first N bytes from that file, using the AID-ALP (TX) message
- The server awaits a AID-ALP (ACK) or AID-ALP (NAK) message
- If can then continue, sending the next N bytes if the message was acknowledged
- Once all data has been transferred, or a NAK has been received, the server sends an AID-ALP (STOP) message

Note that:

- N should not be larger than ~700 bytes (due to the input buffers on the RS232/USB lines). Smaller values of N might improve reliability
- N must be a multiple of 2
- There is no re-send mechanism; if a NAK message is received, the full downloading process must be restarted
- There is no explicit checksum, but an implicit one, as the ALP file already includes a checksum to verify consistency

Overview of the different versions of AID-ALP messages

Short Name	*Content*	*Direction*
AID-ALP (TX)	ALP server sends Data to client	Server -> Client
AID-ALP (STOP)	ALP server terminates a transfer sequence	Server -> Client
AID-ALP (ACK)	ALP client acknowledges successful receipt of data.	Client -> Server

Overview of the different versions of AID-ALP messages continued

Short Name	Content	Direction
AID-ALP (NAK)	ALP client indicates a failed reception of data	Client -> Server
AID-ALP (STAT)	ALP client reports status of the ALP data stored in flash memory	Client -> Server

13.7.2 Host-based AlmanacPlus Overview

This functionality is only supported from u-blox 5 Firmware 4.0 and above.

All three versions of AID-ALPSRV messages are used for the case where the storage of an ALP file is not within the receiver's Flash memory, but on the host, and where the host needs to repeatedly deliver data to the GPS receiver. This allows support of the AlmanacPlus functionality for GPS receivers which do not have Flash memory. For messaging details of an implementation where the data is to reside in the receiver's Flash memory, see Flash-based AlmanacPlus Overview

In the following, the GPS receiver is called the **client**, as it primarily requests data, and the host CPU where the ALP file is located in its entirety is called the **server**.

The operation is such that the client sends periodic data requests (the ALP client requests ALPSRV-REQ) to the host, and the host should answer them accordingly, as described below at ALPSRV-SRV

For this mechanism to work, the AID-ALPSRV *message needs to be activated using the normal* CFG-MSG *commands. If it is not activated, no requests are sent out.*

The client may attempt to modify the data which is stored on the server, using the ALPSRV-CLI message. The server can safely ignore such a request, in case the ALP file cannot be modified. However, for improved performance for consecutive receiver restarts, it is recommended to modify the data.

Overview of the three versions of AID-ALPSRV messages

Short Name	Content	Direction
ALPSRV-REQ	ALP client requests AlmanacPlus data from server	Client -> Server
ALPSRV-SRV	ALP server sends AlmanacPlus data to client	Server -> Client
ALPSRV-CLI	ALP client sends AlmanacPlus data to server.	Client -> Server

13.7.3 Message specifics

The three variants of this message always have a header and variable-size data appended within the same message. The first field, `idSize` gives the number of bytes where the header within the UBX payload ends and data starts.

In case of the ALP client request, the server must assemble a new message according to the AID-ALPSRV-SRV variant. The header needs to be duplicated for as many as `idSize` bytes. Additionally, the server needs to fill in the `fileId` and `dataSize` fields. Appended to the `idSize`-sized header, data must be added as requested by the client (from offset `ofs`, for `size` number of values).

13.7.3.1 Range checks

The server needs to perform an out-of-bounds check on the `ofs` (offsets) and `size` fields, as the client may request data beyond the actually available data. If the client request is within the bounds of available data, the `dataSize` field needs to be filled in with 2 x the content of the size field (the `size` field is in units of 16 bits, whereas the `dataSize` field expects number of bytes). If the client request would request data beyond the limits of the buffer, the data should be reduced accordingly, and this actual number of bytes sent shall be indicated in the dataSize field

13.7.3.2 Changing ALP files

The server function periodically attempts to receive new ALP data from an upstream server, as the result of an HTTP request or other means of file transfer.

In case a new file becomes available, the server shall indicate this to the Client. This is the function of the `fileId` field.

The server should number ALP files it serves arbitrarily. The only requirement is that the `fileId` actually is changed when a new file is being served, and that it does not change as long as the same file is being changed.

If the client, as a result of a client request, receives a `fileId` different from the one in earlier requests' replies, it will reinitialize the ALP engine and request data anew.

Further, if the client attempts to send data to the server, using the `ALPSRV-CLI` method, it indicates, which `fileId` needs to be written. The server shall ignore that request in case the `fileId` numbers do not match.

13.7.3.3 Sample Code

u-blox makes available sample code, written in C language, showing a server implementation, serving ALP data from its file system to a client. Please contact your nearest u-blox Field Application Engineer to receive a copy.

13.8 AssistNow Autonomous

This functionality is only supported by u-blox 6 Firmware 7.01 and above.

13.8.1 Introduction

The A-GPS 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 A-GPS without the need for a host and a connection. Based on a broadcast ephemeris downloaded from the satellite (or obtained by A-GPS) 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.

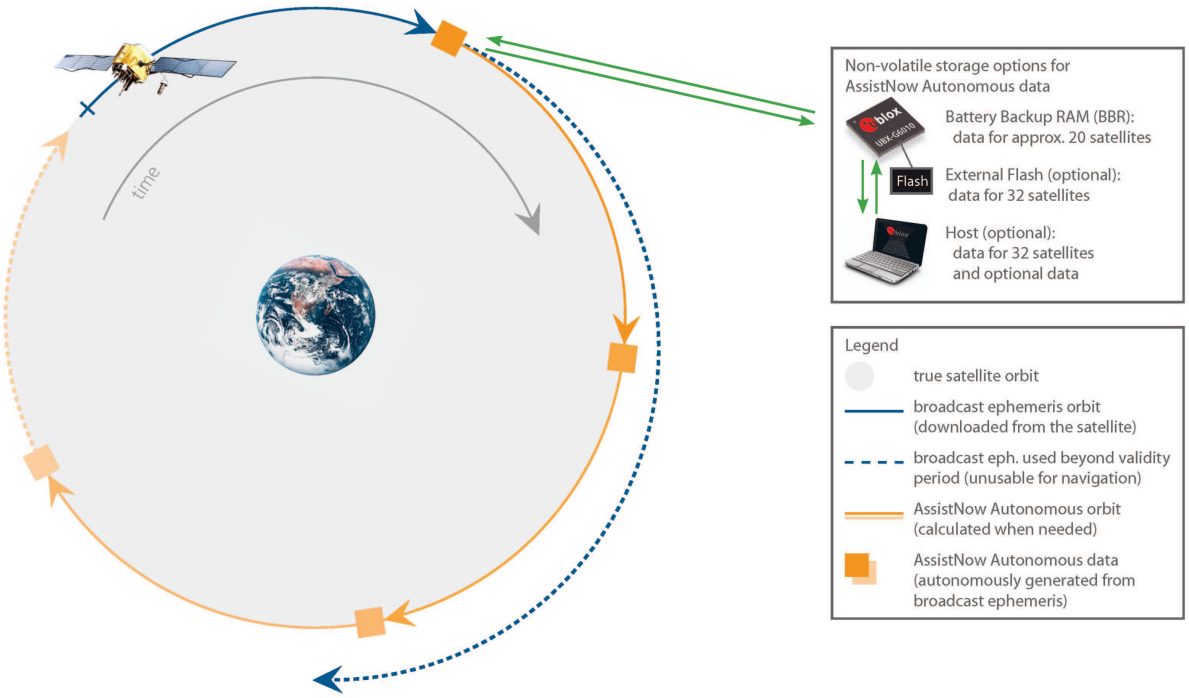
The *AssistNow Autonomous* feature is disabled by default.

13.8.2 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 (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 a broadcast ephemeris. 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. Data for approximately twenty satellites is stored automatically in the on-chip battery backup RAM. Optionally, data for the full constellation (32 satellites) can be stored in external flash memory (if available) or on the host.

- 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).



13.8.3 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 an orbit data validity of approximately three days.
- The `UBX-NAV-AOPSTATUS` message provides information on the current state of the *AssistNow Autonomous* subsystem as well as on the availability of *AssistNow Autonomous* data for individual GPS satellites. The status indicates whether the feature is enabled, and if it is enabled, whether the *AssistNow Autonomous* subsystem is currently idle 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-SVINFO` message indicates the use of *AssistNow Autonomous* orbits for individual satellites.

Two means to preserve *AssistNow Autonomous* data in power-off mode where no battery backup is available are provided:

- The UBX-AID-AOP message provides the host interface to read *AssistNow Autonomous* data from and store it back to the receiver. Note that it is recommended to also read and store GPS almanac data using the UBX-AID-ALM message for best performance. Note that this message can contain additional (optional) data that is not stored in the battery backup RAM or on external flash due to space limitations. This additional data helps the receiver to carry out some calculations faster than without it. It does not, however, affect the orbit quality. Hence, the optional data may be stripped from the message payload if, for example, host storage capacity is limited. Furthermore, it is recommended to use high baud rates on serial interfaces when polling and sending this message due to its relatively large size. Sending (a) valid **UBX-AID-AOP** message(s), to the receiver will automatically enable the *AssistNow Autonomous* feature.
- The UBX-CFG-NVS message provides a means to instruct the receiver to store all available *AssistNow Autonomous* data to an external flash memory. Upon start-up the receiver automatically reads this data if available and merges it with the data from the battery backup RAM (if available). Note that it is recommended to also save GPS almanac data for best performance.

Note that the receiver requires an approximate value of the absolute time to calculate *AssistNow Autonomous* orbits. For best performance it is, therefore, recommended to supply this information to the receiver using the UBX-AID-INI message in a scenario without a running RTC (i.e. without backup battery).

13.8.4 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-GPS). It is, however, required that the receiver roughly know the absolute time, either from an RTC or from time-aiding using the UBX-AID-INI message, 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 vehicle, 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 days. The UBX-CFG-NAVX5 (see above) message allows to change 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] * 38*.

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 satellits. 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 ephemeris extension system 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 GPS satellite constellation is a potential pitfall for the use of the *AssistNow Autonomous* feature. For a given location on Earth the 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 *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 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.

Note that the *AssistNow Autonomous* subsystem will not produce any data and orbits while *AssistNow Offline* data is available.

14 Precise Point Positioning

Please note that this functionality is only supported by u-blox 6 Firmware 7.01 and above.

14.1 Introduction

Precise Point Positioning (PPP) is a premium feature which offers enhanced positioning accuracy by utilizing the carrier phase measurements to smooth the pseudoranges measured to the satellites. The algorithm needs continuous carrier phase measurements to be able to smooth the pseudorange measurements effectively. Additionally ionospheric corrections like those received from SBAS or from GPS are required. A positioning improvement can only be expected in an environment with unobstructed sky view during a period on the order of minutes.

Best results are achieved by combining the PPP algorithm with valuable SBAS corrections.

14.2 Configuration

In order to use the *Precise Point Positioning* algorithm, PPP must be enabled by setting the appropriate flag in UBX-CFG-NAVX5.

PPP can only be activated if the Premium Feature Precise Point Positioning is available.

14.3 Monitoring

The message UBX-NAV-SVINFO indicates for each satellite in use whether or not the pseudorange has been smoothed by the PPP algorithm.

15 Automotive Dead Reckoning (ADR)

15.1 Introduction

u-blox GPS solutions for Automotive Dead Reckoning (ADR) allow high accuracy positioning for various applications at places with poor or no GPS coverage. This technology relies on additional inputs from external sensors measuring the motion of the platform.

ADR is based on Sensor Fusion Dead Reckoning (SFDR) technology using sensors external to the GPS receiver.

The following solutions / sets of sensors are supported being combined with GPS:

- Gyro and single wheel tick (GWT) solution

The ADR solution uses the messages of the External Sensor Fusion (ESF) class.

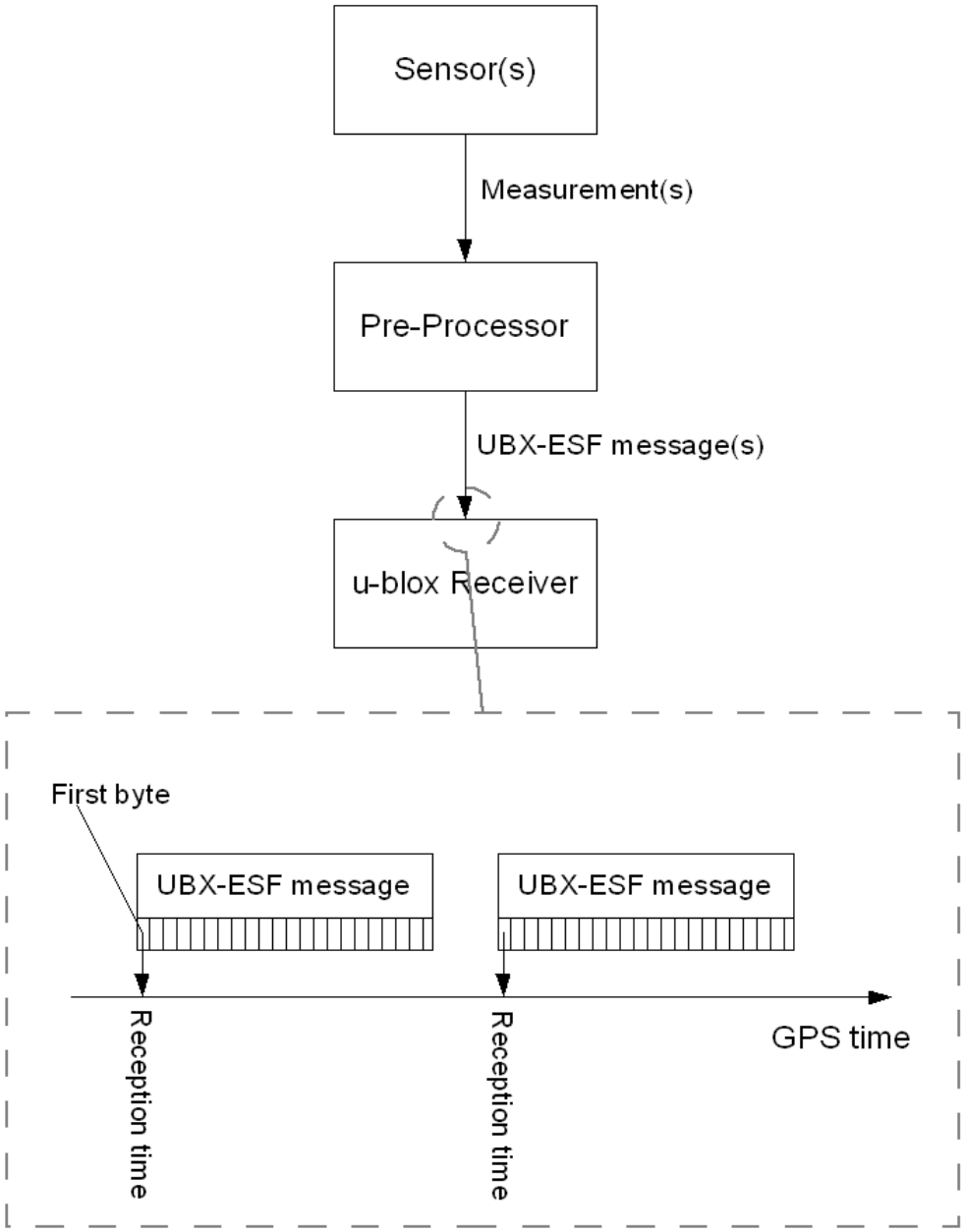
15.2 Timing

Knowing the timing of the external sensor measurements in the GPS receiver time frame is essential for achieving optimal performance with the ADR-based navigation solution. Sensor timing must be precisely aligned with the GPS receiver time frame. There are different ways to reduce the latency of the external sensor measurements, and to solve the timing of the external sensor measurements in relation to the GPS receiver time frame:

- First Byte Reception: reception time of first byte of ESF-MEAS message
- Time Mark on External Input: time mark signal on external input

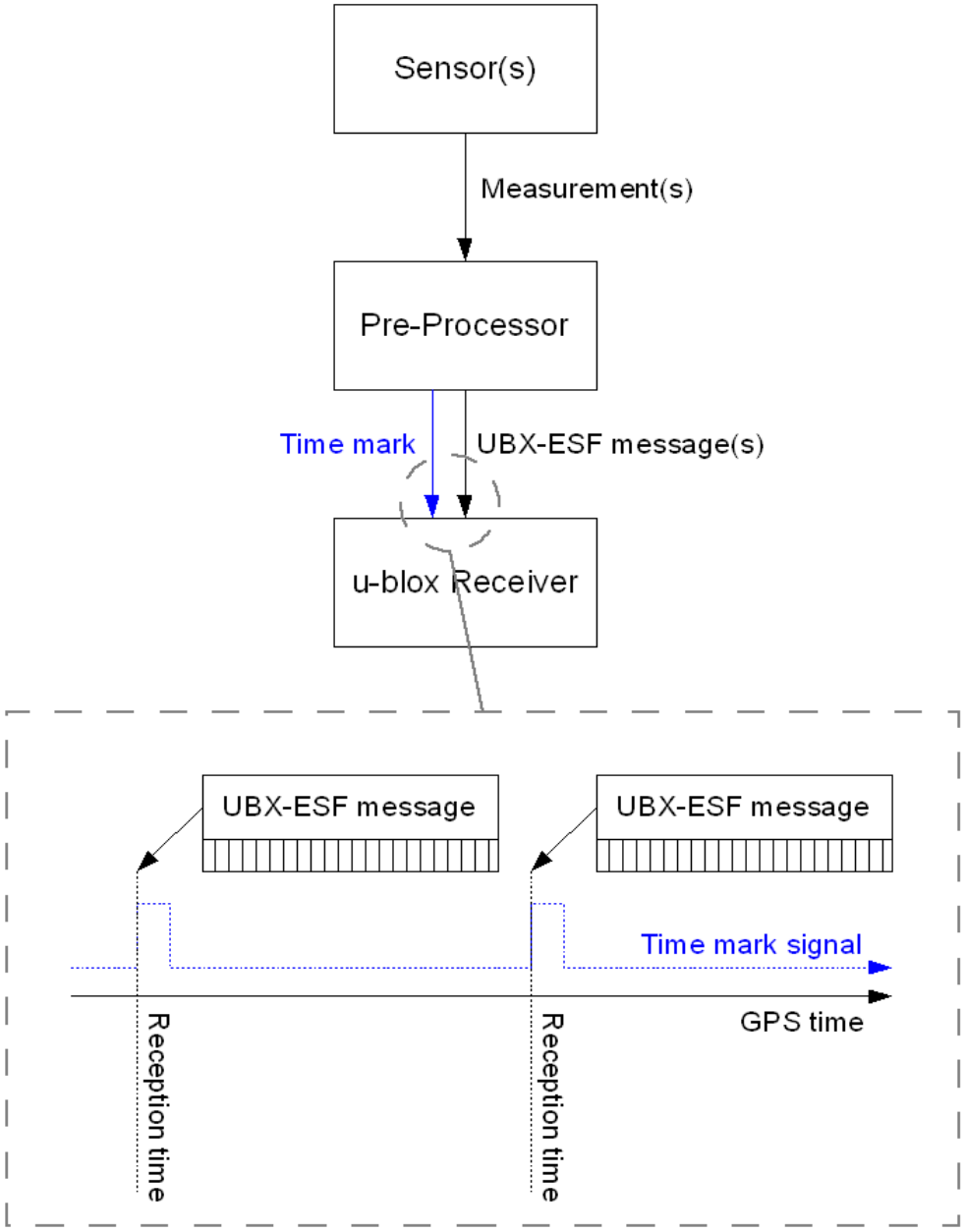
15.2.1 First Byte Reception

The easiest way to solve the issue is to have the GPS receiver assume the time of the reception of the first byte of the ESF-MEAS message (minus a constant latency) to be the time of sensor measurement. This approach is the simplest to implement, but the next approach can yield better latency control and compensation.



15.2.2 Time Mark on External Input

In this case, the preprocessor unit generating the measurements sends a signal to the EXTINT input of the GPS receiver, marking the moment of measurement generation. The subsequently following ESF-MEAS message is then flagged accordingly. The time of the signal reception will then be attributed to the measurement values contained in the message. This approach is the preferred solution, but it can be difficult to realize an exact analog time signal for the preprocessor unit.



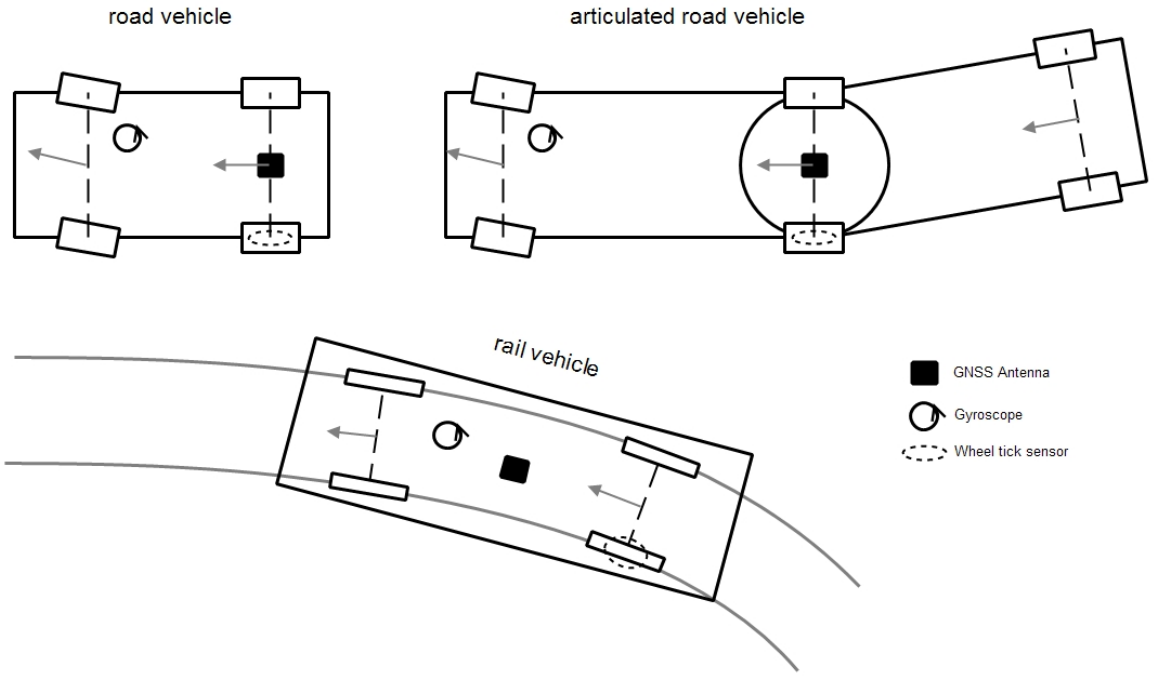
15.2.3 Latency

Depending on the timing approach chosen, the latency of the sensor data corresponds to the time period between the point when the sensor measurement was taken and the detection of either *the first byte* of the ESF-MEAS message or the pre-processor's *time mark* at the receiver.

15.3 Setup recommendations

For an optimum ADR navigation performance, the following setup recommendations should be considered.

15.3.1 GPS antenna placement, gyro placement and single tick origin



Due to geometric and dynamic aspects of driving vehicles, it is important to correctly place the GPS antenna and the external sensors - from a geometric point of view - in order to get consistent measurement information from the different sensors.

For **standard road vehicles** (no articulation): The GPS antenna should be placed above the middle of the rear (unsteered) axis. The gyro can be placed anywhere on the vehicle. Single ticks should origin from the rear (unsteered) wheels.

For **articulated vehicles**, the sensors should be placed on the front car as if this was a standard road vehicle.

In case the GWT solution is used for **rail vehicles**: The GPS antenna should be placed in the middle of a wagon, while the gyro can be placed anywhere on the same wagon and the single ticks can origin from any wheels of the same wagon.

Large geometrical deviations from the optimum placement - especially of the GPS antenna (e.g. when placing it above the front axis of a long bus) - can result in significant performance degradations!

15.3.2 Startup/Shutdown integration guideline

Continuous dead reckoning performance is possible if

- NVS storage is available (i.e., BBR is actually battery backed)
- the sensor data stream is only started/stopped while the vehicle is not moving
- the vehicle is not moved or turned while the receiver is off

In general, when the last sensor information was that the vehicle is moving, the receiver switches to GPS-only navigation during periods of external sensor data unavailability.

15.3.3 Navigation and measurement rate recommendations

For an optimum ADR navigation performance, the standard navigation rate of 1 Hz and sensor measurement inputs with frequencies of 10 Hz are recommended.

The wheel tick quantization error is a limiting factor when using high frequency updates. This means that navigation rates higher than 1 Hz may result in lower position accuracies.

15.4 ESF Measurement Data (LEA-6R)

The ESF-MEAS message is used to provide external sensor data to the ESF Dead Reckoning solution. In ESF-MEAS a variable number of data fields are available which can contain various types of measurements. The type of each measurement in a data field is defined as follows:

Definition of data types

Type	*Description*	*Unit*	*Format of the 24 data bits*
0	none, data field contains no data		
1..4	reserved		
5	gyro reading vertical axis	deg/s *2^-12	signed
6..9	reserved		
10	single tick (speed tick)		Bits 0-22: unsigned tick value. Bit 23: direction indicator (0=forwards, 1=backwards)
11	reserved		
12	temperature	deg celsius * 1e-2	signed
13..255	reserved		

LEA-6R module only processes single tick (speed tick) measurements. Any other type of measurements contained in ESF-MEAS is ignored.

15.5 Gyro and Wheel Tick (GWT) Solution Configuration (LEA-6R)

15.5.1 Attached Gyroscope and Analog Wheel Ticks

u-blox Dead Reckoning GPS solutions based on Gyroscope and Wheel Ticks (GWT) allow high accuracy positioning for automotive applications at places with poor or no GPS coverage. This technology relies on additional inputs of a gyroscope and a speed sensor providing heading rate and wheel tick measurements. Optionally an additional temperature sensor can be used to continuously compensate temperature-dependent

gyroscope measurement errors.

For GWT based dead reckoning certain sensor parameters need to be defined. The UBX configuration messages CFG-EKF is used to set those **mandatory** parameters:

- `inverseFlags`: invert meaning of the gyroscope rotation sense
- `nomZero`: nominal gyroscope zero point output
- `nomSens`: nominal gyroscope sensitivity

If the drift of the gyroscope bias should be compensated by means of a *zero velocity temperature compensation* the following parameters are **mandatory**:

- `rmsTemp`: maximum allowable RMS threshold for zero velocity temperature compensation
- `tempUpdate`: time interval in which the temperature compensation table is saved to non-volatile storage

The temperature compensation RMS threshold depends on the gyroscope noise and the environmental conditions (vibrations) and must be set appropriately.

Optional parameters are:

- `nomPPDist`: nominal pulses per distance
- `pulsesPerM`: nominal pulses per distance is given in pulses per m instead of pulses per km

If the nominal pulses per distance is not defined a coarse a-priori value is estimated before starting the sensor calibration.

15.5.2 Using Serial Wheel Ticks

Instead of an analog signal the wheel tick measurements can be received in ESF-MEAS messages via serial port. The `useSerWt` flag has to be set accordingly in CFG-EKF. In addition, the time and wheel tick parameters are **mandatory** and must be defined using CFG-ESFGWT:

- `timeTagFactor`: sensor time tag factor
- `timeTagMax`: maximum value of sensor time tag
- `wtCountMax`: maximum value of tick counter
- `wtLatency`: latency of wheel tick data
- `wtFrequency`: nominal wheel tick data frequency

In connection with wheel tick (WT) measurements it will be distinguished between relative and absolute wheel ticks which are defined as follows:

Definition of relative and absolute WT measurements

Name	*Description*
absolute WT count	continuous count of wheel ticks since start up at ttag = 0
relative WT count	count of wheel ticks between a certain time period

The time period `dt` belonging to the relative wheel tick count is calculated from its ttag and the ttag of the previous measurement.

It is strongly recommended to use absolute wheel ticks in order to ensure a robust measurement processing even after sensor failures or outages.

Absolute wheel ticks are always counted continuously regardless of driving forwards or backwards. Driving direction is indicated separately (see description of ESF measurement data).

The ttag is always expected to be a continuous ttag since start up at ttag = 0.

The latency of the sensor data should be given as accurate as possible to achieve best positioning performance. The minimum accuracy should be at least 10ms. The data frequency is used to initialize the data base and should be known with an accuracy of about 10%.

The following parameters of the CFG-ESFGWT message are **optional**:

- `wtFactor`: wheel tick factor
- `wtQuantError`: wheel tick quantization error

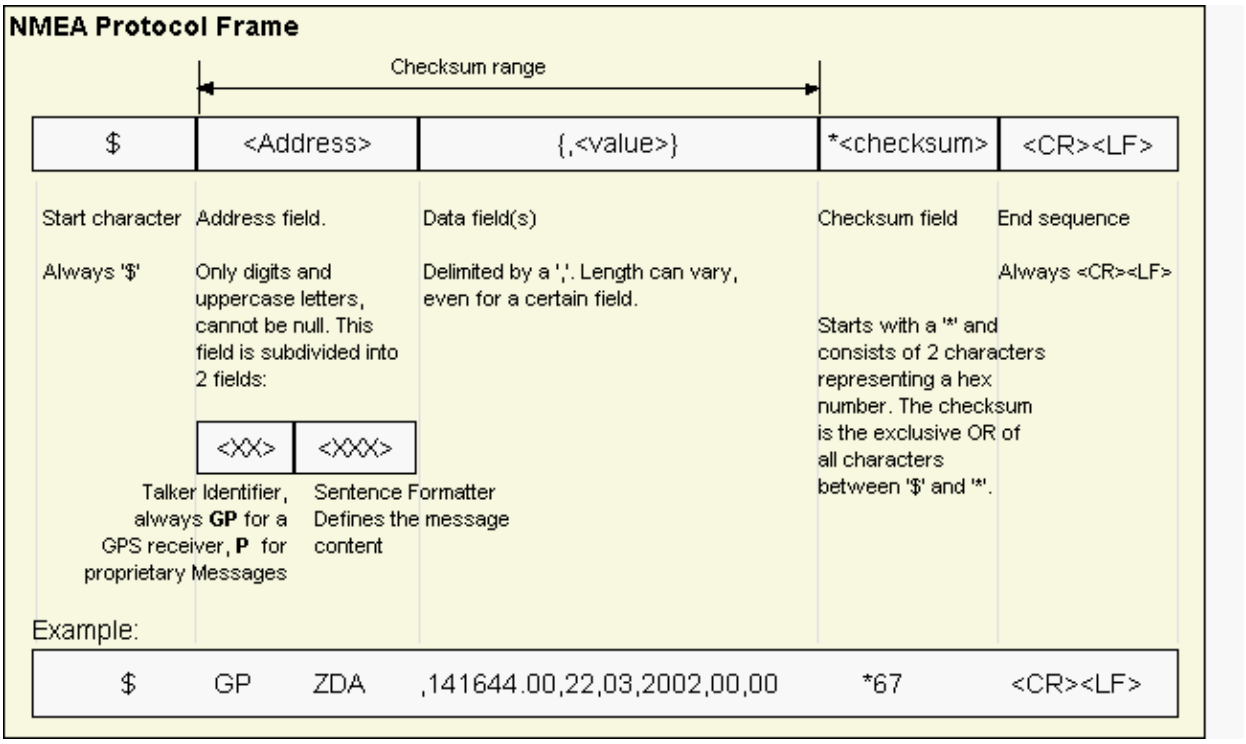
If the tick factor `wtFactor` is not given, it will be estimated by the receiver. This estimation can take up to several minutes, depending on the receiver dynamics and the quality of the GPS solution. Once determined, the tick factor will be stored to the non-volatile storage so that it will be immediately available after a restart. The quantization error `wtQuantError` only needs to be set if the tick measurement does not contain raw tick counts (e.g., if the tick measurement is in fact a distance).

If the optional parameters are previously known it is recommended to configure them to avoid possible estimation inaccuracies.

NMEA Protocol

16 Protocol Overview

NMEA messages sent by the GPS receiver are based on NMEA 0183 Version 2.3. The following picture shows the structure of a NMEA protocol message.



For further information on the NMEA Standard please refer to *NMEA 0183 Standard For Interfacing Marine Electronic Devices*, Version 2.30, March 1, 1998. 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.

17 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

18 Position Fix Flags in NMEA Mode

The following list shows how u-blox implements the NMEA protocol, and the conditions determining how flags are set in version 2.3 and above.

NMEA Message: Field	*No position fix (at power-up, after losing satellite lock)*	*Valid position fix with GPS, but user limits exceeded*	*Valid dead reckoning fix, but user limits exceeded*	*Dead reckoning (linear extrapolation, ADR with external sensors, or map matching)*	*2D position fix*	*3D position fix*	*combined GPS/SFDR position fix (ADR with external sensors)*
GLL, RMC: Status	V	V	V	A	A	A	A
	A=Data VALID, V=Data Invalid (Navigation Receiver Warning)						
GGA: Quality Indicator	0	0	6	6	1 / 2	1 / 2	1 / 2
	0=Fix not available/invalid, 1=GPS SPS Mode, Fix valid, 2=Differential GPS, SPS Mode, Fix Valid, 6=Estimated/Dead Reckoning						
GSA: Nav Mode	1	1	2	2	2	3	3
	1=Fix Not available, 2=2D Fix, 3=3D Fix						
GLL, RMC, VTG, GNS: Mode Indicator	N	N	E	E	A / D	A / D	A / D
	N=No Fix, A=Autonomous GNSS Fix, D=Differential GNSS Fix, E=Estimated/Dead Reckoning Fix						
UBX GPSFixOK	0	0	0	1	1	1	1
UBX GPSFix	0	>1	1	1	2	3	4

The following list shows how u-blox implements the NMEA protocol, and the conditions determining how flags are set in version 2.2 and below.

NMEA Message: Field	*No position fix (at power-up, after losing satellite lock*	*Valid position fix with GPS, but user limits exceeded*	*Valid dead reckoning fix, but user limits exceeded*	*Dead reckoning (linear extrapolation, ADR with external sensors, or map matching)*	*2D position fix*	*3D position fix*	*combined GPS/SFDR position fix (ADR with external sensors)*
GLL, RMC: Status	V	V	V	A	A	A	A
	A=Data VALID, V=Data Invalid (Navigation Receiver Warning)						
GGA: Quality Indicator	0	0	1	1	1 / 2	1 / 2	1 / 2
	0=Fix not available/invalid, 1=GPS SPS Mode, Fix valid, 2=Differential GPS, SPS Mode, Fix Valid						
GSA: Nav Mode	1	1	2	2	2	3	3
	1=Fix Not available, 2=2D Fix, 3=3D Fix						
GLL, RMC, VTG: Mode Indicator. This field is not output by this NMEA version.							
GNS: This message is not defined in this NMEA version.							
UBX GPSFixOK	0	0	0	1	1	1	1
UBX GPSFix	0	>1	1	1	2	3	4

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
```

An exception from the above default are dead reckoning fixes, which are also output when invalid (user limits exceeded).

In Antaris firmware versions older than 3.0, the receiver did output invalid data and marked it with the 'Invalid/Valid' Flags. If required, this function can still be enabled in later firmware versions, using the UBX protocol message CFG-NMEA.

Differing from the NMEA standard, u-blox reports valid dead reckoning fixes with user limits met (not exceeded) as valid (A) instead of invalid (V).

19 NMEA Messages Overview

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

Page	*Mnemonic*	*Cls/ID*	*Description*
	NMEA Proprietary Messages		**Proprietary Messages**
68	**UBX,00**	0xF1 0x00	Poll a PUBX,00 message
69	**UBX,00**	0xF1 0x00	Lat/Long Position Data
71	**UBX,03**	0xF1 0x03	Poll a PUBX,03 message
72	**UBX,03**	0xF1 0x03	Satellite Status
74	**UBX,04**	0xF1 0x04	Poll a PUBX,04 message
75	**UBX,04**	0xF1 0x04	Time of Day and Clock Information
76	**UBX,05**	0xF1 0x05	Poll a PUBX,05 message
77	**UBX,05**	0xF1 0x05	Lat/Long Position Data
79	**UBX,06**	0xF1 0x06	Poll a PUBX,06 message
80	**UBX,06**	0xF1 0x06	Lat/Long Position Data
82	**UBX,40**	0xF1 0x40	Set NMEA message output rate
83	**UBX,41**	0xF1 0x41	Set Protocols and Baudrate
	NMEA Standard Messages		**Standard Messages**
54	**DTM**	0xF0 0x0A	Datum Reference
55	**GBS**	0xF0 0x09	GNSS Satellite Fault Detection
56	**GGA**	0xF0 0x00	Global positioning system fix data
57	**GLL**	0xF0 0x01	Latitude and longitude, with time of position fix and status
58	**GPQ**	0xF0 0x40	Poll message
59	**GRS**	0xF0 0x06	GNSS Range Residuals
60	**GSA**	0xF0 0x02	GNSS DOP and Active Satellites
61	**GST**	0xF0 0x07	GNSS Pseudo Range Error Statistics
62	**GSV**	0xF0 0x03	GNSS Satellites in View
63	**RMC**	0xF0 0x04	Recommended Minimum data
64	**THS**	0xF0 0x0E	True Heading and Status
65	**TXT**	0xF0 0x41	Text Transmission
66	**VTG**	0xF0 0x05	Course over ground and Ground speed

NMEA Messages Overview continued

Page	*Mnemonic*	*Cls/ID*	*Description*
67	**ZDA**	0xF0 0x08	Time and Date

20 Standard Messages

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

20.1 DTM

Message	**DTM**		
Description	**Datum Reference**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	This message gives the difference between the currently selected Datum, and the reference Datum. If the currently configured Datum is not WGS84 or WGS72, then the field **LLL** will be set to 999, and the field **LSD** is set to a variable-length string, representing the Name of the Datum. The list of supported datums can be found in CFG-DAT. The reference Datum can not be changed and is always set to WGS84.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x0A	11	

Message Structure:

```
$GPDTM,LLL,LSD,lat,N/S,lon,E/W,alt,RRR*cs<CR><LF>
```

Example:

```
$GPDTM,W84,,0.0,N,0.0,E,0.0,W84*6F
$GPDTM,W72,,0.00,S,0.01,W,-2.8,W84*4F
$GPDTM,999,CH95,0.08,N,0.07,E,-47.7,W84*1C
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPDTM	string	`$GPDTM`	-	Message ID, DTM protocol header
1	W72	string	`LLL`	-	Local Datum Code, W84 = WGS84, W72 = WGS72, 999 = user defined
2	-	string	`LSD`	-	Local Datum Subdivision Code, This field outputs the currently selected Datum as a string (see also note above).
3	0.08	numeric	`lat`	minutes	Offset in Latitude
4	S	character	`NS`	-	North/South indicator
5	0.07	numeric	`lon`	minutes	Offset in Longitude
6	E	character	`EW`	-	East/West indicator
7	-2.8	numeric	`alt`	m	Offset in altitude
8	W84	string	`RRR`	-	Reference Datum Code, W84 = WGS 84. This is the only supported Reference datum.
9	*67	hexadecimal	`cs`	-	Checksum
10	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.2 GBS

Message	**GBS**		
Description	**GNSS Satellite Fault Detection**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
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 this case - 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.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x09	11	

Message Structure:

```
$GPGBS,hhmmss.ss,errlat,errlon,erralt,svid,prob,bias,stddev*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*5B
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGBS	string	`$GPGBS`	-	Message ID, GBS protocol header
1	235503.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Time to which this RAIM sentence belongs
2	1.6	numeric	`errlat`	m	Expected error in latitude
3	1.4	numeric	`errlon`	m	Expected error in longitude
4	3.2	numeric	`erralt`	m	Expected error in altitude
5	03	numeric	`svid`	-	Satellite ID of most likely failed satellite
6	-	numeric	`prob`	-	Probability of missed detection, no supported (empty)
7	-21.4	numeric	`bias`	m	Estimate on most likely failed satellite (a priori residual)
8	3.8	numeric	`stddev`	m	Standard deviation of estimated bias
9	*40	hexadecimal	`cs`	-	Checksum
10	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.3 GGA

Message	**GGA**		
Description	**Global positioning system fix data**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	**The output of this message is dependent on the currently selected datum (Default: WGS84)** 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:

```
$GPGGA,hhmmss.ss,Latitude,N,Longitude,E,FS,NoSV,HDOP,msl,m,Altref,m,DiffAge,DiffStation*cs<CR><LF>
```

Example:

```
$GPGGA,092725.00,4717.11399,N,00833.91590,E,1,8,1.01,499.6,M,48.0,M,,0*5B
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGGA	string	`$GPGGA`	-	Message ID, GGA protocol header
1	092725.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Current time
2	4717.11399	ddmm.mmmm	`Latitude`	-	Latitude, Degrees + minutes, see Format description
3	N	character	`N`	-	N/S Indicator, N=north or S=south
4	00833.91590	dddmm.mmmm	`Longitude`	-	Longitude, Degrees + minutes, see Format description
5	E	character	`E`	-	E/W indicator, E=east or W=west
6	1	digit	`FS`	-	Position Fix Status Indicator, See Table below and Position Fix Flags description
7	8	numeric	`NoSV`	-	Satellites Used, Range 0 to 12
8	1.01	numeric	`HDOP`	-	HDOP, Horizontal Dilution of Precision
9	499.6	numeric	`msl`	m	MSL Altitude
10	M	character	`uMsl`	-	Units, Meters (fixed field)
11	48.0	numeric	`Altref`	m	Geoid Separation
12	M	character	`uSep`	-	Units, Meters (fixed field)
13	-	numeric	`DiffAge`	s	Age of Differential Corrections, Blank (Null) fields when DGPS is not used
14	0	numeric	`DiffStation`	-	Diff. Reference Station ID
15	*5B	hexadecimal	`cs`	-	Checksum
16	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

Table Fix Status

Fix Status	*Description, see also Position Fix Flags description*
0	No Fix / Invalid
1	Standard GPS (2D/3D)
2	Differential GPS
6	Estimated (DR) Fix

20.4 GLL

Message	**GLL**		
Description	**Latitude and longitude, with time of position fix and status**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
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	(9) or (10)	

Message Structure:

```
$GPGLL,Latitude,N,Longitude,E,hhmmss.ss,Valid,Mode*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGLL	string	`$GPGLL`	-	Message ID, GLL protocol header
1	4717.11364	ddmm.mmmm	`Latitude`	-	Latitude, Degrees + minutes, see Format description
2	N	character	`N`	-	N/S Indicator, hemisphere N=north or S=south
3	00833.91565	dddmm.mmmm	`Longitude`	-	Longitude, Degrees + minutes, see Format description
4	E	character	`E`	-	E/W indicator, E=east or W=west
5	092321.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Current time
6	A	character	`Valid`	-	V = Data invalid or receiver warning, A = Data valid. See Position Fix Flags description
Start of optional block					
7	A	character	`Mode`	-	Positioning Mode, see Position Fix Flags description
End of optional block					
7	*60	hexadecimal	`cs`	-	Checksum
8	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.5 GPQ

Message	**GPQ**		
Description	**Poll message**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Input Message		
Comment	Polls a standard NMEA message.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x40	4	

Message Structure:

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

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$EIGPQ	string	`$xxGPQ`	-	Message ID, GPQ protocol header, xx = talker identifier
1	RMC	string	`sid`	-	Sentence identifier
2	*3A	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.6 GRS

Message	**GRS**		
Description	**GNSS Range Residuals**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
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.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x06	17	

Message Structure:

```
$GPGRS,hhmmss.ss, mode {,residual}*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,,,*70
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGRS	string	`$GPGRS`	-	Message ID, GRS protocol header
1	082632.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Time of associated position fix
2	1	digit	`mode`	-	Mode (see table below), u-blox receivers will always output Mode 1 residuals
Start of repeated block (12 times)					
3 + 1*N	0.54	numeric	`residual`	m	Range residuals for SVs used in navigation. The SV order matches the order from the GSA sentence.
End of repeated block					
15	*70	hexadecimal	`cs`	-	Checksum
16	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

Table Mode

Mode	*Description*
0	Residuals were used to calculate the position given in the matching GGA sentence.
1	Residuals were recomputed after the GGA position was computed.

20.7 GSA

Message	**GSA**		
Description	**GNSS DOP and Active Satellites**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	The GPS 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)		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x02	20	

Message Structure:

```
$GPGSA,Smode,FS{,sv},PDOP,HDOP,VDOP*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGSA	string	`$GPGSA`	-	Message ID, GSA protocol header
1	A	character	`Smode`	-	Smode, see first table below
2	3	digit	`FS`	-	Fix status, see second table below and Position Fix Flags description
Start of repeated block (12 times)					
3 + 1*N	29	numeric	`sv`	-	Satellite number
End of repeated block					
15	1.94	numeric	`PDOP`	-	Position dilution of precision
16	1.18	numeric	`HDOP`	-	Horizontal dilution of precision
17	1.54	numeric	`VDOP`	-	Vertical dilution of precision
18	*0D	hexadecimal	`cs`	-	Checksum
19	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

Table Smode

Smode	*Description*
M	Manual - forced to operate in 2D or 3D mode
A	Allowed to automatically switch 2D/3D mode

Table Fix Status

Fix Status	*Description, see also Position Fix Flags description*
1	Fix not available
2	2D Fix
3	3D Fix

20.8 GST

Message	**GST**		
Description	**GNSS Pseudo Range Error Statistics**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	-		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x07	11	

Message Structure:

```
$GPGST,hhmmss.ss,range_rms,std_major,std_minor,hdg,std_lat,std_long,std_alt*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGST	string	`$GPGST`	-	Message ID, GST protocol header
1	082356.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Time of associated position fix
2	1.8	numeric	`range_rms`	m	RMS value of the standard deviation of the ranges
3	-	numeric	`std_major`	m	Standard deviation of semi-major axis, not supported (empty)
4	-	numeric	`std_minor`	m	Standard deviation of semi-minor axis, not supported (empty)
5	-	numeric	`hdg`	degrees	Orientation of semi-major axis, not supported (empty)
6	1.7	numeric	`std_lat`	m	Standard deviation of latitude, error in meters
7	1.3	numeric	`std_long`	m	Standard deviation of longitude, error in meters
8	2.2	numeric	`std_alt`	m	Standard deviation of altitude, error in meters
9	*7E	hexadecimal	`cs`	-	Checksum
10	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.9 GSV

Message	**GSV**		
Description	**GNSS Satellites in View**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	The number of satellites in view, together with each PRN (SV ID), elevation and azimuth, and C/No (Signal/Noise Ratio) value. Only four satellite details are transmitted in one message.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x03	7..16	

Message Structure:

```
$GPGSV,NoMsg,MsgNo,NoSv,{,sv,elv,az,cno}*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPGSV	string	`$GPGSV`	-	Message ID, GSV protocol header
1	3	digit	`NoMsg`	-	Number of messages, total number of GPGSV messages being output
2	1	digit	`MsgNo`	-	Number of this message
3	10	numeric	`NoSv`	-	Satellites in View
Start of repeated block (1..4 times)					
4 + 4*N	23	numeric	`sv`	-	Satellite ID
5 + 4*N	38	numeric	`elv`	degrees	Elevation, range 0..90
6 + 4*N	230	numeric	`az`	degrees	Azimuth, range 0..359
7 + 4*N	44	numeric	`cno`	dBHz	C/N0, range 0..99, null when not tracking
End of repeated block					
5..16	*7F	hexadecimal	`cs`	-	Checksum
6..16	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.10 RMC

Message	**RMC**		
Description	**Recommended Minimum data**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	**The output of this message is dependent on the currently selected datum (Default: WGS84)** The Recommended Minimum sentence defined by NMEA for GPS/Transit system data.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x04	15	

Message Structure:

```
$GPRMC,hhmmss,status,latitude,N,longitude,E,spd,cog,ddmmyy,mv,mvE,mode*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPRMC	string	`$GPRMC`	-	Message ID, RMC protocol header
1	083559.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Time of position fix
2	A	character	`Status`	-	Status, V = Navigation receiver warning, A = Data valid, see Position Fix Flags description
3	4717.11437	ddmm.mmmm	`Latitude`	-	Latitude, Degrees + minutes, see Format description
4	N	character	`N`	-	N/S Indicator, hemisphere N=north or S=south
5	00833.91522	dddmm.mmmm	`Longitude`	-	Longitude, Degrees + minutes, see Format description
6	E	character	`E`	-	E/W indicator, E=east or W=west
7	0.004	numeric	`Spd`	knots	Speed over ground
8	77.52	numeric	`Cog`	degrees	Course over ground
9	091202	ddmmyy	`date`	-	Date in day, month, year format
10	-	numeric	`mv`	degrees	Magnetic variation value, not being output by receiver
11	-	character	`mvE`	-	Magnetic variation E/W indicator, not being output by receiver
12	-	character	`mode`	-	Mode Indicator, see Position Fix Flags description
13	*57	hexadecimal	`cs`	-	Checksum
14	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.11 THS

Message	**THS**		
Description	**True Heading and Status**		
Firmware	Supported on u-blox 6 firmware version 7.03 (**only available with ADR product variant**).		
Type	Output Message		
Comment	Actual vehicle heading in degrees, true heading. This message is only available on LEA-6R variant version 2.02.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x0E	5	

Message Structure:

```
$GPTHS,headt,status*cs<CR><LF>
```

Example:

```
$GPTHS,77.52,E*32
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPTHS	string	`$GPTHS`	-	Message ID, THS protocol header
1	77.52	numeric	`headt`	degrees	Heading of vehicle (true)
2	E	character	`mi`	-	Mode indicator: A = autonomous, E = Estimated (dead reckoning), M = Manual input, S = Simulator, V = Data not valid
3	*32	hexadecimal	`cs`	-	Checksum
4	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.12 TXT

Message	**TXT**		
Description	**Text Transmission**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	**This message is not configured through CFG-MSG, but instead through 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 CFG-INF		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x41	7	

Message Structure:

```
$GPTXT,xx,yy,zz,ascii data*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.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPTXT	string	`$GPTXT`	-	Message ID, TXT protocol header
1	01	numeric	`xx`	-	Total number of messages in this transmission, 01..99
2	01	numeric	`yy`	-	Message number in this transmission, range 01..xx
3	02	numeric	`zz`	-	Text identifier, u-blox GPS receivers specify the severity of the message with this number. - 00 = ERROR - 01 = WARNING - 02 = NOTICE - 07 = USER
4	www.u-blox.com	string	`string`	-	Any ASCII text
5	*67	hexadecimal	`cs`	-	Checksum
6	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.13 VTG

Message	**VTG**		
Description	**Course over ground and Ground speed**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	Velocity is given as Course over Ground (COG) and Speed over Ground (SOG).		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x05	12	

Message Structure:

```
$GPVTG,cogt,T,cogm,M,sog,N,kph,K,mode*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPVTG	string	`$GPVTG`	-	Message ID, VTG protocol header
1	77.52	numeric	`cogt`	degrees	Course over ground (true)
2	T	character	`T`	-	Fixed field: true
3	-	numeric	`cogm`	degrees	Course over ground (magnetic), not output
4	M	character	`M`	-	Fixed field: magnetic
5	0.004	numeric	`sog`	knots	Speed over ground
6	N	character	`N`	-	Fixed field: knots
7	0.008	numeric	`kph`	km/h	Speed over ground
8	K	character	`K`	-	Fixed field: kilometers per hour
9	A	character	`mode`	-	Mode Indicator, see Position Fix Flags description
10	*06	hexadecimal	`cs`	-	Checksum
11	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

20.14 ZDA

Message	**ZDA**		
Description	**Time and Date**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	-		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF0 0x08	9	

Message Structure:

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

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$GPZDA	string	`$GPZDA`	-	Message ID, ZDA protocol header
1	082710.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time
2	16	dd	`day`	day	UTC time: day, 01..31
3	09	mm	`month`	month	UTC time: month, 01..12
4	2002	yyyy	`year`	year	UTC time: 4 digit year
5	00	-xx	`ltzh`	-	Local zone hours, not supported (fixed to 00)
6	00	zz	`ltzn`	-	Local zone minutes, not supported (fixed to 00)
7	*64	hexadecimal	`cs`	-	Checksum
8	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21 Proprietary Messages

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

21.1 UBX,00

Message	**UBX,00**		
Description	**Poll a PUBX,00 message**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Input Message		
Comment	A PUBX,00 message is polled by sending the PUBX,00 message without any data fields.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x00	4	

Message Structure:

```
$PUBX,00*33<CR><LF>
```

Example:

```
$PUBX,00*33
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	00	numeric	`MsgID`	-	Set to 00 to poll a PUBX,00 message
2	*33	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.2 UBX,00

Message	**UBX,00**		
Description	**Lat/Long Position Data**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
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 CFG-DAT.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x00	23	

Message Structure:

```
$PUBX,00,hhmmss.ss,Latitude,N,Longitude,E,AltRef,NavStat,Hacc,Vacc,SOG,COG,Vvel,ageC,HDOP,VDOP,TDOP
,GU,RU,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,,0.92,1.19,0.7
7,9,0,0*5F
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	00	numeric	`ID`	-	Proprietary message identifier: 00
2	081350.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Current time
3	4717.113210	ddmm.mmmm	`Latitude`	-	Latitude, Degrees + minutes, see Format description
4	N	character	`N`	-	N/S Indicator, N=north or S=south
5	00833.915187	dddmm.mmmm	`Longitude`	-	Longitude, Degrees + minutes, see Format description
6	E	character	`E`	-	E/W indicator, E=east or W=west
7	546.589	numeric	`AltRef`	m	Altitude above user datum ellipsoid.
8	G3	string	`NavStat`	-	Navigation Status, See Table below
9	2.1	numeric	`Hacc`	m	Horizontal accuracy estimate.
10	2.0	numeric	`Vacc`	m	Vertical accuracy estimate.
11	0.007	numeric	`SOG`	km/h	Speed over ground
12	77.52	numeric	`COG`	degrees	Course over ground
13	0.007	numeric	`Vvel`	m/s	Vertical velocity, positive=downwards
14	-	numeric	`ageC`	s	Age of most recent DGPS corrections, empty = none available
15	0.92	numeric	`HDOP`	-	HDOP, Horizontal Dilution of Precision
16	1.19	numeric	`VDOP`	-	VDOP, Vertical Dilution of Precision
17	0.77	numeric	`TDOP`	-	TDOP, Time Dilution of Precision
18	9	numeric	`GU`	-	Number of GPS satellites used in the navigation solution

UBX,00 continued

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
19	0	numeric	RU	-	Number of GLONASS satellites used in the navigation solution
20	0	numeric	DR	-	DR used
21	*5B	hexadecimal	cs	-	Checksum
22	-	character	<CR><LF>	-	Carriage Return and Line Feed

Table Navigation Status

Navigation Status	*Description*
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

21.3 UBX,03

Message	**UBX,03**		
Description	**Poll a PUBX,03 message**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Input Message		
Comment	A PUBX,03 message is polled by sending the PUBX,03 message without any data fields.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x03	4	

Message Structure:

```
$PUBX,03*30<CR><LF>
```

Example:

```
$PUBX,03*30
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	03	numeric	`MsgID`	-	Set to 03 to poll a PUBX,03 message
2	*30	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.4 UBX,03

Message	**UBX,03**		
Description	**Satellite Status**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	The PUBX,03 message contains satellite status information.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x03	5 + 6*GT	

Message Structure:

```
$PUBX,03,GT{,SVID,s,AZM,EL,SN,LK},*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.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	03	numeric	`ID`	-	Proprietary message identifier: 03
2	11	numeric	`GT`	-	Number of GPS satellites tracked
Start of repeated block (GT times)					
3 + 6*N	23	numeric	`SVID`	-	Satellite PRN number
4 + 6*N	-	character	`s`	-	Satellite status, see table below
5 + 6*N	-	numeric	`AZM`	degrees	Satellite azimuth, range 000..359
6 + 6*N	-	numeric	`EL`	degrees	Satellite elevation, range 00..90
7 + 6*N	45	numeric	`SN`	dBHz	Signal to noise ratio, range 00..55
8 + 6*N	010	numeric	`LK`	s	Satellite carrier lock time, range 00..64 0 = code lock only 64 = lock for 64 seconds or more
End of repeated block					
3 + 6*GT	*0D	hexadecimal	`cs`	-	Checksum
4 + 6*GT	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

Table Satellite Status

Satellite Status	*Description*
-	Not used
U	Used in solution
e	Ephemeris available, but not used for navigation

21.5 UBX,04

Message	**UBX,04**		
Description	**Poll a PUBX,04 message**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Input Message		
Comment	A PUBX,04 message is polled by sending the PUBX,04 message without any data fields.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x04	4	

Message Structure:

```
$PUBX,04*37<CR><LF>
```

Example:

```
$PUBX,04*37
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	04	numeric	`MsgID`	-	Set to 04 to poll a PUBX,04 message
2	*37	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.6 UBX,04

Message	**UBX,04**		
Description	**Time of Day and Clock Information**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Output Message		
Comment	-		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x04	12	

Message Structure:

```
$PUBX,04,hhmmss.ss,ddmmyy,UTC_TOW,UTC_WNO,LEAP_SEC,Clk_B,Clk_D,PG,*cs<CR><LF>
```

Example:

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

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	04	numeric	`ID`	-	Proprietary message identifier: 04
2	073731.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Current time in hour, minutes, seconds
3	091202	ddmmyy	`ddmmyy`	-	UTC Date, day, month, year format
4	113851.00	numeric	`UTC_TOW`	s	UTC Time of Week
5	1196	numeric	`UTC_WNO`	-	UTC week number, continues beyond 1023
6	15D	numeric/text	`LEAP_SEC`	s	Before FW 7.01: reserved. FW 7.01 and above: Leap seconds, The number is marked with a 'D' if the value is the firmware default value (15 for FW 7.00). If the value is not marked it has been received from a satellite.
7	1930035	numeric	`Clk_B`	ns	Receiver clock bias
8	-2660.664	numeric	`Clk_D`	ns/s	Receiver clock drift
9	43	numeric	`PG`	ns	Timepulse Granularity, The quantization error of the Timepulse pin
10	*3C	hexadecimal	`cs`	-	Checksum
11	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.7 UBX,05

Message	**UBX,05**		
Description	**Poll a PUBX,05 message**		
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).		
Type	Input Message		
Comment	A PUBX,05 message is polled by sending the PUBX,05 message without any data fields.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x05	4	

Message Structure:

```
$PUBX,05*36<CR><LF>
```

Example:

```
$PUBX,05*36
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	05	numeric	`MsgID`	-	Set to 05 to poll a PUBX,05 message
2	*36	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.8 UBX,05

Message	**UBX,05**		
Description	**Lat/Long Position Data**		
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).		
Type	Output Message		
Comment	**This message is only provided for backwards compatibility and should not be utilized for future designs.** -		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x05	19	

Message Structure:

```
$PUBX,05,,*cs<CR><LF>
```

Example:

```
$PUBX,06,,0*5F
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	05	numeric	`ID`	-	Proprietary message identifier: 05
2	1346	numeric	`pulses`	-	Number of pulses in last time period [0-9999]
3	1000	numeric	`period`	ms	Duration of last time period [0-9999]
4	32424	numeric	`gyroMean`	-	Uncorrected average Gyro value in last period [0-65535]
5	17.8	numeric	`temperature`	°C	Temperature
6	F	character	`direction`	-	Forward(F)/Backward(B) Indicator
7	3	numeric	`pulseScaleCS`	-	Calibration status of speed pulse scale factor (see table below)
8	2	numeric	`gyroScaleCS`	-	Calibration status of gyroscope scale factor (see table below)
9	3	numeric	`gyroBiasCS`	-	Calibration status of gyroscope bias (see table below)
10	0.0171	numeric	`pulseScale`	-	Current scale factor of speed pulse
11	0.00323	numeric	`gyroBias`	rad/s	Current gyroscope bias
12	0.998	numeric	`gyroScale`	-	Current gyroscope scale factor
13	94	numeric	`pulseScaleAcc`	%	Accuracy of speed pulse scale factor in percentage of initial value
14	98	numeric	`gyroBiasAcc`	%	Accuracy of gyroscope bias in percentage of initial value
15	97	numeric	`gyroScaleAcc`	%	Accuracy of gyroscope scale factor in percentage of initial value
16	0F	hexadecimal	`measUsed`	-	Measurements used (see table below)

UBX,05 continued

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
17	*0D	hexadecimal	cs	-	Checksum
18	-	character	<CR><LF>	-	Carriage Return and Line Feed

Table Sensor Calibration Status

Sensor Calibration Status	*Description*
0	no calibration
1	calibrating
2	coarse calibration
3	fine calibration

Table Measurements used

Measurements used	*Description*
Bit 0	Speed pulse used
Bit 1	forward/backward signal used
Bit 2	Gyroscope used
Bit 3	Temperature used
Bit 4	GPS position used
Bit 5	GPS velocity used
Bit 6	Inconsitency with the gyroscope sensor input detected. Sensor Fusion temporarily disabled. GPS-only data being output.
Bit 7	Inconsitency with the speed pulse sensor input detected. Sensor Fusion temporarily disabled. GPS-only data being output.

21.9 UBX,06

Message	**UBX,06**		
Description	**Poll a PUBX,06 message**		
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).		
Type	Input Message		
Comment	A PUBX,06 message is polled by sending the PUBX,06 message without any data fields.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x06	4	

Message Structure:

```
$PUBX,06*35<CR><LF>
```

Example:

```
$PUBX,06*35
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	06	numeric	`MsgID`	-	Set to 06 to poll a PUBX,06 message
2	*35	hexadecimal	`cs`	-	Checksum
3	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.10 UBX,06

Message	**UBX,06**		
Description	**Lat/Long Position Data**		
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).		
Type	Output Message		
Comment	**This message is only provided for backwards compatibility and should not be utilized for future designs.** -		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	0xF1 0x06	23	

Message Structure:

```
$PUBX,06,hhmmss.ss,Latitude,N,Longitude,E,AltRef,NavStat,Hacc,Vacc,SOG,COG,Vvel,ageC,HDOP,VDOP,TDOP
,GU,RU,DR,*cs<CR><LF>
```

Example:

```
$PUBX,06,081350.00,4717.113210,N,00833.915187,E,546.589,G3,2.1,2.0,0.007,77.52,0.007,,0.92,1.19,0.7
7,9,0,0*5F
```

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	06	numeric	`ID`	-	Proprietary message identifier: 06
2	081350.00	hhmmss.sss	`hhmmss.ss`	-	UTC Time, Current time
3	4717.113210	ddmm.mmmm	`Latitude`	-	Latitude, Degrees + minutes, see Format description
4	N	character	`N`	-	N/S Indicator, N=north or S=south
5	00833.915187	dddmm.mmmm	`Longitude`	-	Longitude, Degrees + minutes, see Format description
6	E	character	`E`	-	E/W indicator, E=east or W=west
7	546.589	numeric	`AltRef`	m	Altitude above user datum ellipsoid.
8	G3	string	`NavStat`	-	Navigation Status, See Table below
9	2.1	numeric	`Hacc`	m	Horizontal accuracy estimate.
10	2.0	numeric	`Vacc`	m	Vertical accuracy estimate.
11	0.007	numeric	`SOG`	km/h	Speed over ground
12	77.52	numeric	`COG`	degrees	Course over ground
13	0.007	numeric	`Vvel`	m/s	Vertical velocity, positive=downwards
14	-	numeric	`ageC`	s	Age of most recent DGPS corrections, empty = none available
15	0.92	numeric	`HDOP`	-	HDOP, Horizontal Dilution of Precision
16	1.19	numeric	`VDOP`	-	VDOP, Vertical Dilution of Precision
17	0.77	numeric	`TDOP`	-	TDOP, Time Dilution of Precision
18	9	numeric	`GU`	-	Number of GPS satellites used in the navigation solution
19	0	numeric	`RU`	-	Number of GLONASS satellites used in the navigation solution

UBX,06 continued

Field No.	*Example*	*Format*	*Name*	*Unit*	*Description*
20	0	numeric	`reserved`	-	
21	*0D	hexadecimal	`cs`	-	Checksum
22	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

Table Navigation Status

Navigation Status	*Description*
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

21.11 UBX,40

Message	**UBX,40**		
Description	**Set NMEA message output rate**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Set Message		
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.		
Message Info	*ID for CFG-MSG*	*Number of fields*	
	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.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	40	numeric	`ID`	-	Proprietary message identifier
2	GLL	string	`MsgId`	-	NMEA message identifier
3	1	numeric	`rddc`	cycles	output rate on DDC - 0 disables that message from being output on this port - 1 means that this message is output every epoch
4	1	numeric	`rus1`	cycles	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	1	numeric	`rus2`	cycles	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	1	numeric	`rusb`	cycles	output rate on USB - 0 disables that message from being output on this port - 1 means that this message is output every epoch
7	1	numeric	`rspi`	cycles	output rate on SPI - 0 disables that message from being output on this port - 1 means that this message is output every epoch
8	0	numeric	`reserved`	-	Reserved, Always fill with 0
9	*5D	hexadecimal	`cs`	-	Checksum
10	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

21.12 UBX,41

Message	**UBX,41**		
Description	**Set Protocols and Baudrate**		
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.		
Type	Set Message		
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.	*Example*	*Format*	*Name*	*Unit*	*Description*
0	$PUBX	string	`$PUBX`	-	Message ID, UBX protocol header, proprietary sentence
1	41	numeric	`ID`	-	Proprietary message identifier
2	1	numeric	`portID`	-	ID of communication port, for a list of port IDs see CFG-PRT.
3	0007	hexadecimal	`inProto`	-	Input protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in CFG-PRT.
4	0003	hexadecimal	`outProto`	-	Output protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in CFG-PRT.
5	19200	numeric	`baudrate`	bits/s	Baudrate
6	0	numeric	`autobauding`	-	Autobauding: 1=enable, 0=disable (not supported on u-blox 5, set to 0)
7	*25	hexadecimal	`cs`	-	Checksum
8	-	character	`<CR><LF>`	-	Carriage Return and Line Feed

UBX Protocol

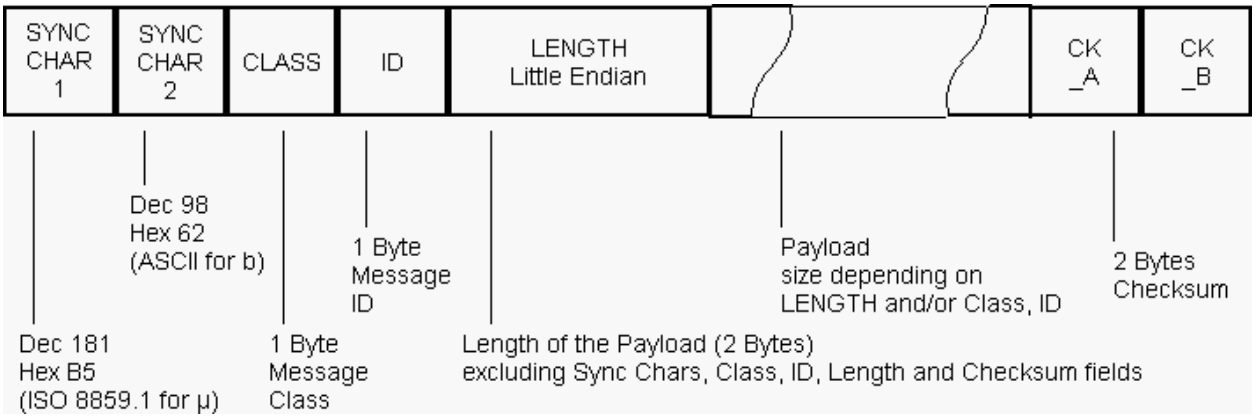
22 UBX Protocol Key Features

u-blox GPS receivers use a u-blox proprietary protocol to transmit GPS data to a host computer using asynchronous RS232 ports. 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)

23 UBX Packet Structure

A basic UBX Packet looks as follows:



- Every Message starts with 2 Bytes: 0xB5 0x62
- A 1 Byte Class Field follows. The Class defines the basic subset of the message
- A 1 Byte ID Field defines the message that is to follow
- A 2 Byte Length Field is following. Length is defined as being the length of the payload, only. It does not include Sync Chars, Length Field, Class, ID or CRC fields. The number format of the length field is an unsigned 16-Bit integer in Little Endian Format.
- The Payload is a variable length field.
- CK_A and CK_B is a 16 Bit checksum whose calculation is defined below.

24 UBX Class IDs

A Class is a grouping of messages which are related to each other. The following table gives the short names, description and Class ID Definitions.

Name	*Class*	*Description*
NAV	0x01	Navigation Results: Position, Speed, Time, Acc, 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/Nack Messages: as replies to CFG Input Messages
CFG	0x06	Configuration Input Messages: Set Dynamic Model, Set DOP Mask, Set Baud Rate, etc.
MON	0x0A	Monitoring Messages: Comunication Status, CPU Load, Stack Usage, Task Status
AID	0x0B	AssistNow Aiding Messages: Ephemeris, Almanac, other A-GPS data input
TIM	0x0D	Timing Messages: Timepulse Output, Timemark Results

UBX Class IDs continued

Name	Class	Description
ESF	0x10	External Sensor Fusion Messages: External sensor measurements and status information

All remaining class IDs are reserved.

25 UBX Payload Definition Rules

25.1 Structure Packing

Values are placed in an order that structure packing is not a problem. This means that 2Byte values shall start on offsets which are a multiple of 2, 4-byte values shall start at a multiple of 4, and so on. This can easily be achieved by placing the largest values first in the Message payload (e.g. R8), and ending with the smallest (i.e. one-byters such as U1) values.

25.2 Message Naming

Referring to messages is done by adding the class name and a dash in front of the message name. For example, the ECEF-Message is referred to as NAV-POSECEF. Referring to values is done by adding a dash and the name, e.g. NAV-POSECEF-X

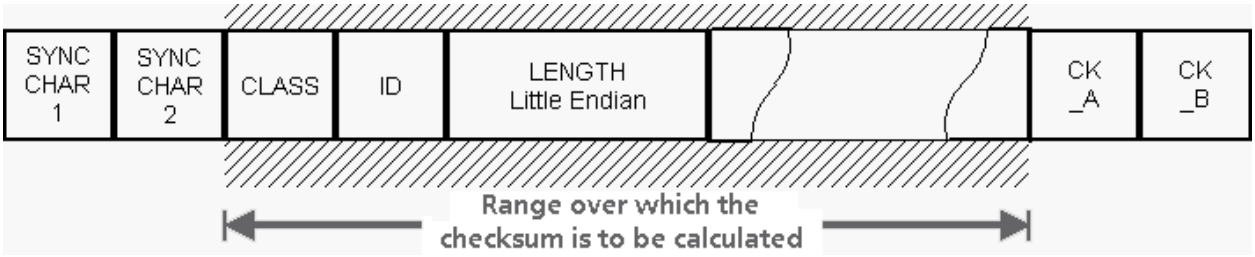
25.3 Number Formats

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. A technical description of the IEEE754 format can be found in the AnswerBook from the ADS1.x toolkit.
The following table gives information about the various values:

Short	Type	Size (Bytes)	Comment	Min/Max	Resolution
U1	Unsigned Char	1		0..255	1
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
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			

26 UBX Checksum

The checksum is calculated over the packet, 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 at the end of the packet.

27 UBX Message Flow

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

27.1 Acknowledgement

When messages from the Class CFG are sent to the receiver, the receiver will send an Acknowledge (ACK-ACK) or a Not Acknowledge (ACK-NAK) message back to the sender, depending on whether or not the message was processed correctly.
There is no ACK/NAK mechanism for message poll requests outside Class CFG.

27.2 Polling Mechanism

All messages that are output by the receiver in a periodic manner (i.e. Messages in Classes MON, NAV and RXM) can also be polled.
There is not a single specific message which polls any other message. The UBX protocol was designed such, that when sending a message with no payload (or just a single parameter which identifies the poll request) the message is polled.

28 UBX Messages Overview

Page	Mnemonic	Cls/ID	Length	Type	Description
UBX Class ACK				**Ack/Nack Messages**	
91	**ACK-ACK**	0x05 0x01	2	Answer	Message Acknowledged
91	**ACK-NAK**	0x05 0x00	2	Answer	Message Not-Acknowledged
UBX Class AID				**AssistNow Aiding Messages**	
92	**AID-ALM**	0x0B 0x30	0	Poll Request	Poll GPS Aiding Almanac Data
92	**AID-ALM**	0x0B 0x30	1	Poll Request	Poll GPS Aiding Almanac Data for a SV
93	**AID-ALM**	0x0B 0x30	(8) or (40)	Input/Output Message	GPS Aiding Almanac Input/Output Message
93	**AID-ALPSRV**	0x0B 0x32	16	Output Message	ALP client requests AlmanacPlus data from server
94	**AID-ALPSRV**	0x0B 0x32	16 + 1*dataSize	Input Message	ALP server sends AlmanacPlus data to client
95	**AID-ALPSRV**	0x0B 0x32	8 + 2*size	Output Message	ALP client sends AlmanacPlus data to server.
95	**AID-ALP**	0x0B 0x50	0 + 2*N	Input message	ALP file data transfer to the receiver
96	**AID-ALP**	0x0B 0x50	1	Input message	Mark end of data transfer
96	**AID-ALP**	0x0B 0x50	1	Output message	Acknowledges a data transfer
96	**AID-ALP**	0x0B 0x50	1	Output message	Indicate problems with a data transfer
97	**AID-ALP**	0x0B 0x50	24	Periodic/Polled	Poll the AlmanacPlus status
97	**AID-AOP**	0x0B 0x33	0	Poll request	Poll AssistNow Autonomous data
98	**AID-AOP**	0x0B 0x33	1	Poll request	Poll AssistNow Autonomous data for one satellite
98	**AID-AOP**	0x0B 0x33	(48) or (192)	Input/Output Message	AssistNow Autonomous data
99	**AID-DATA**	0x0B 0x10	0	Poll	Polls all GPS Initial Aiding Data
99	**AID-EPH**	0x0B 0x31	0	Poll Request	Poll GPS Aiding Ephemeris Data
99	**AID-EPH**	0x0B 0x31	1	Poll Request	Poll GPS Aiding Ephemeris Data for a SV
100	**AID-EPH**	0x0B 0x31	(8) or (104)	Input/Output Message	GPS Aiding Ephemeris Input/Output Message
101	**AID-HUI**	0x0B 0x02	0	Poll Request	Poll GPS Health, UTC and ionosphere parameters
101	**AID-HUI**	0x0B 0x02	72	Input/Output Message	GPS Health, UTC and ionosphere parameters
102	**AID-INI**	0x0B 0x01	0	Poll Request	Poll GPS Initial Aiding Data
103	**AID-INI**	0x0B 0x01	48	Polled	Aiding position, time, frequency, clock drift
104	**AID-REQ**	0x0B 0x00	0	Virtual	Sends a poll (AID-DATA) for all GPS Aiding Data
UBX Class CFG				**Configuration Input Messages**	
105	**CFG-ANT**	0x06 0x13	0	Poll Request	Poll Antenna Control Settings
105	**CFG-ANT**	0x06 0x13	4	Get/Set	Get/Set Antenna Control Settings
106	**CFG-CFG**	0x06 0x09	(12) or (13)	Command	Clear, Save and Load configurations
108	**CFG-DAT**	0x06 0x06	0	Poll Request	Poll Datum Setting
108	**CFG-DAT**	0x06 0x06	2	Set	Set Standard Datum
108	**CFG-DAT**	0x06 0x06	44	Set	Set User-defined Datum
109	**CFG-DAT**	0x06 0x06	52	Get	Get currently selected Datum
110	**CFG-EKF**	0x06 0x12	0	Poll Request	Poll EKF Module Settings
110	**CFG-EKF**	0x06 0x12	16	Get/Set	Get/Set EKF Module Settings - LEA-6R

UBX Messages Overview continued

Page	*Mnemonic*	*Cls/ID*	*Length*	*Type*	*Description*
112	**CFG-ESFGWT**	0x06 0x29	44	Get/Set message	Get/Set settings of gyro+wheel tick sol (GWT) - LEA-6R
113	**CFG-FXN**	0x06 0x0E	0	Poll Request	Poll FXN configuration
113	**CFG-FXN**	0x06 0x0E	36	Command	RXM FixNOW configuration.
114	**CFG-INF**	0x06 0x02	1	Poll Request	Poll INF message configuration for one protocol
115	**CFG-INF**	0x06 0x02	0 + 10*N	Set/Get	Information message configuration
116	**CFG-ITFM**	0x06 0x39	8	Command	Jamming/Interference Monitor configuration.
117	**CFG-MSG**	0x06 0x01	2	Poll Request	Poll a message configuration
117	**CFG-MSG**	0x06 0x01	8	Set/Get	Set Message Rate(s)
118	**CFG-MSG**	0x06 0x01	3	Set/Get	Set Message Rate
118	**CFG-NAV5**	0x06 0x24	0	Poll Request	Poll Navigation Engine Settings
119	**CFG-NAV5**	0x06 0x24	36	Get/Set	Get/Set Navigation Engine Settings
120	**CFG-NAVX5**	0x06 0x23	0	Poll Request	Poll Navigation Engine Expert Settings
120	**CFG-NAVX5**	0x06 0x23	40	Get/Set	Get/Set Navigation Engine Expert Settings
122	**CFG-NMEA**	0x06 0x17	0	Poll Request	Poll the NMEA protocol configuration
122	**CFG-NMEA**	0x06 0x17	4	Set/Get	Set/Get the NMEA protocol configuration
123	**CFG-NVS**	0x06 0x22	13	Command	Clear, Save and Load non-volatile storage data
125	**CFG-PM2**	0x06 0x3B	0	Poll Request	Poll extended Power Management configuration
125	**CFG-PM2**	0x06 0x3B	44	Set/Get	Extended Power Management configuration
127	**CFG-PM**	0x06 0x32	0	Poll Request	Poll Power Management configuration
127	**CFG-PM**	0x06 0x32	24	Set/Get	Power Management configuration
129	**CFG-PRT**	0x06 0x00	0	Poll Request	Polls the configuration of the used I/O Port
129	**CFG-PRT**	0x06 0x00	1	Poll Request	Polls the configuration for one I/O Port
129	**CFG-PRT**	0x06 0x00	20	Get/Set	Get/Set Port Configuration for UART
132	**CFG-PRT**	0x06 0x00	20	Get/Set	Get/Set Port Configuration for USB Port
133	**CFG-PRT**	0x06 0x00	20	Get/Set	Get/Set Port Configuration for SPI Port
136	**CFG-PRT**	0x06 0x00	20	Get/Set	Get/Set Port Configuration for DDC Port
138	**CFG-RATE**	0x06 0x08	0	Poll Request	Poll Navigation/Measurement Rate Settings
138	**CFG-RATE**	0x06 0x08	6	Get/Set	Navigation/Measurement Rate Settings
139	**CFG-RINV**	0x06 0x34	0	Poll Request	Poll contents of Remote Inventory
139	**CFG-RINV**	0x06 0x34	1 + 1*N	Set/Get	Set/Get contents of Remote Inventory
140	**CFG-RST**	0x06 0x04	4	Command	Reset Receiver / Clear Backup Data Structures
141	**CFG-RXM**	0x06 0x11	0	Poll Request	Poll RXM configuration
141	**CFG-RXM**	0x06 0x11	2	Set/Get	RXM configuration
142	**CFG-SBAS**	0x06 0x16	0	Poll Request	Poll contents of SBAS Configuration
142	**CFG-SBAS**	0x06 0x16	8	Command	SBAS Configuration
144	**CFG-TMODE2**	0x06 0x3D	0	Poll Request	Poll Time Mode Settings
144	**CFG-TMODE2**	0x06 0x3D	28	Get/Set	Time Mode Settings 2
145	**CFG-TMODE**	0x06 0x1D	0	Poll Request	Poll Time Mode Settings

UBX Messages Overview continued

Page	*Mnemonic*	*Cls/ID*	*Length*	*Type*	*Description*
145	**CFG-TMODE**	0x06 0x1D	28	Get/Set	Time Mode Settings
146	**CFG-TP5**	0x06 0x31	0	Poll Request	Poll Timepulse Parameters
146	**CFG-TP5**	0x06 0x31	1	Poll Request	Poll TimePulse Parameters
147	**CFG-TP5**	0x06 0x31	32	Get/Set	Get/Set TimePulse Parameters
148	**CFG-TP**	0x06 0x07	0	Poll Request	Poll TimePulse Parameters
148	**CFG-TP**	0x06 0x07	20	Get/Set	Get/Set TimePulse Parameters
149	**CFG-USB**	0x06 0x1B	0	Poll Request	Poll a USB configuration
149	**CFG-USB**	0x06 0x1B	108	Get/Set	Get/Set USB Configuration
UBX Class ESF				**External Sensor Fusion Messages**	
151	**ESF-MEAS**	0x10 0x02	(8 + 4*N) or (12 + 4*N)	Input/Output Message	External Sensor Fusion Measurements (LEA-6R)
152	**ESF-STATUS**	0x10 0x10	16 + 4*numSens	Periodic/Polled	Sensor Fusion Status Information (LEA-6R)
154	**ESF-STATUS**	0x10 0x10	16 + 4*numSens	Periodic/Polled	Sensor Fusion Status Information (LEA-6R)
UBX Class INF				**Information Messages**	
157	**INF-DEBUG**	0x04 0x04	0 + 1*N	Output	ASCII String output, indicating debug output
157	**INF-ERROR**	0x04 0x00	0 + 1*N	Output	ASCII String output, indicating an error
158	**INF-NOTICE**	0x04 0x02	0 + 1*N	Output	ASCII String output, with informational contents
158	**INF-TEST**	0x04 0x03	0 + 1*N	Output	ASCII String output, indicating test output
159	**INF-WARNING**	0x04 0x01	0 + 1*N	Output	ASCII String output, indicating a warning
UBX Class MON				**Monitoring Messages**	
160	**MON-HW2**	0x0A 0x0B	28	Periodic/Polled	Extended Hardware Status
161	**MON-HW**	0x0A 0x09	68	Periodic/Polled	Hardware Status
162	**MON-HW**	0x0A 0x09	68	Periodic/Polled	Hardware Status
163	**MON-IO**	0x0A 0x02	0 + 20*N	Periodic/Polled	I/O Subsystem Status
164	**MON-MSGPP**	0x0A 0x06	120	Periodic/Polled	Message Parse and Process Status
164	**MON-RXBUF**	0x0A 0x07	24	Periodic/Polled	Receiver Buffer Status
165	**MON-RXR**	0x0A 0x21	1	Get	Receiver Status Information
165	**MON-TXBUF**	0x0A 0x08	28	Periodic/Polled	Transmitter Buffer Status
166	**MON-VER**	0x0A 0x04	70 + 30*N	Answer to Poll	Receiver/Software/ROM Version
UBX Class NAV				**Navigation Results**	
167	**NAV-AOPSTATUS**	0x01 0x60	20	Periodic/Polled	AssistNow Autonomous Status
167	**NAV-CLOCK**	0x01 0x22	20	Periodic/Polled	Clock Solution
168	**NAV-DGPS**	0x01 0x31	16 + 12*numCh	Periodic/Polled	DGPS Data Used for NAV
169	**NAV-DOP**	0x01 0x04	18	Periodic/Polled	Dilution of precision
169	**NAV-EKFSTATUS**	0x01 0x40	36	Periodic/Polled	Dead Reckoning Software Status
171	**NAV-POSECEF**	0x01 0x01	20	Periodic/Polled	Position Solution in ECEF
172	**NAV-POSLLH**	0x01 0x02	28	Periodic/Polled	Geodetic Position Solution
172	**NAV-SBAS**	0x01 0x32	12 + 12*cnt	Periodic/Polled	SBAS Status Data
174	**NAV-SOL**	0x01 0x06	52	Periodic/Polled	Navigation Solution Information

UBX Messages Overview continued

Page	*Mnemonic*	*Cls/ID*	*Length*	*Type*	*Description*
175	**NAV-STATUS**	0x01 0x03	16	Periodic/Polled	Receiver Navigation Status
177	**NAV-SVINFO**	0x01 0x30	8 + 12*numCh	Periodic/Polled	Space Vehicle Information
179	**NAV-TIMEGPS**	0x01 0x20	16	Periodic/Polled	GPS Time Solution
179	**NAV-TIMEUTC**	0x01 0x21	20	Periodic/Polled	UTC Time Solution
180	**NAV-VELECEF**	0x01 0x11	20	Periodic/Polled	Velocity Solution in ECEF
181	**NAV-VELNED**	0x01 0x12	36	Periodic/Polled	Velocity Solution in NED
UBX Class RXM				**Receiver Manager Messages**	
182	**RXM-ALM**	0x02 0x30	0	Poll Request	Poll GPS Constellation Almanach Data
182	**RXM-ALM**	0x02 0x30	1	Poll Request	Poll GPS Constellation Almanach Data for a SV
183	**RXM-ALM**	0x02 0x30	(8) or (40)	Poll Answer / Periodic	GPS Aiding Almanach Input/Output Message
183	**RXM-EPH**	0x02 0x31	0	Poll Request	Poll GPS Constellation Ephemeris Data
184	**RXM-EPH**	0x02 0x31	1	Poll Request	Poll GPS Constellation Ephemeris Data for a SV
184	**RXM-EPH**	0x02 0x31	(8) or (104)	Poll Answer / Periodic	GPS Aiding Ephemeris Input/Output Message
185	**RXM-PMREQ**	0x02 0x41	8	Input	Requests a Power Management task
185	**RXM-RAW**	0x02 0x10	8 + 24*numSV	Periodic/Polled	Raw Measurement Data
186	**RXM-SFRB**	0x02 0x11	42	Periodic	Subframe Buffer
187	**RXM-SVSI**	0x02 0x20	8 + 6*numSV	Periodic/Polled	SV Status Info
UBX Class TIM				**Timing Messages**	
189	**TIM-SVIN**	0x0D 0x04	28	Periodic/Polled	Survey-in data
189	**TIM-TM2**	0x0D 0x03	28	Periodic/Polled	Time mark data
191	**TIM-TP**	0x0D 0x01	16	Periodic/Polled	Timepulse Timedata
192	**TIM-VRFY**	0x0D 0x06	20	Polled/Once	Sourced Time Verification

29 ACK (0x05)

Ack/Nack Messages: i.e. as replies to CFG Input Messages.
Messages in this class are sent as a result of a CFG message being received, decoded and processed by the receiver.

29.1 ACK-ACK (0x05 0x01)

29.1.1 Message Acknowledged

Message	**ACK-ACK**				
Description	**Message Acknowledged**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Answer				
Comment	Output upon processing of an input message				
Message Structure	*Header*	*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

29.2 ACK-NAK (0x05 0x00)

29.2.1 Message Not-Acknowledged

Message	**ACK-NAK**				
Description	**Message Not-Acknowledged**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Answer				
Comment	Output upon processing of an input message				
Message Structure	*Header*	*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

30 AID (0x0B)

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

30.1 AID-ALM (0x0B 0x30)

30.1.1 Poll GPS Aiding Almanac Data

Message	**AID-ALM**				
Description	**Poll GPS Aiding Almanac Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	**This message has an empty payload!** 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*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x30	0	*see below*	CK_A CK_B
No payload					

30.1.2 Poll GPS Aiding Almanac Data for a SV

Message	**AID-ALM**				
Description	**Poll GPS Aiding Almanac Data for a SV**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	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*	*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).

30.1.3 GPS Aiding Almanac Input/Output Message

Message	**AID-ALM**				
Description	**GPS Aiding Almanac Input/Output Message**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input/Output Message				
Comment	• 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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					

30.2 AID-ALPSRV (0x0B 0x32)

30.2.1 ALP client requests AlmanacPlus data from server

Message	**AID-ALPSRV**				
Description	**ALP client requests AlmanacPlus data from server**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output Message				
Comment	This message is sent by the ALP client to the ALP server in order to request data. The given identifier must be prepended to the requested data when submitting the data.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x32	16	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`idSize`	bytes	Identifier size. This data, beginning at message start, must prepend the returned data.

AID-ALPSRV continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
1	U1	-	`type`	-	Requested data type. Must be different from 0xff, otherwise this is not a data request.
2	U2	-	`ofs`	-	Requested data offset [16bit words]
4	U2	-	`size`	-	Requested data size [16bit words]
6	U2	-	`fileId`	-	Unused when requesting data, filled in when sending back the data
8	U2	-	`dataSize`	bytes	Actual data size. Unused when requesting data, filled in when sending back the data.
10	U1	-	`id1`	-	Identifier data
11	U1	-	`id2`	-	Identifier data
12	U4	-	`id3`	-	Identifier data

30.2.2 ALP server sends AlmanacPlus data to client

Message	**AID-ALPSRV**				
Description	**ALP server sends AlmanacPlus data to client**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input Message				
Comment	This message is sent by the ALP server to the ALP client and is usually sent in response to a data request. The server copies the identifier from the request and fills in the dataSize and fileId fields.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x32	16 + 1*dataSize	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`idSize`	bytes	Identifier size
1	U1	-	`type`	-	Requested data type
2	U2	-	`ofs`	-	Requested data offset [16bit words]
4	U2	-	`size`	-	Requested data size [16bit words]
6	U2	-	`fileId`	-	Corresponding ALP file ID, must be filled in by the server!
8	U2	-	`dataSize`	bytes	Actual data contained in this message, must be filled in by the server!
10	U1	-	`id1`	-	Identifier data
11	U1	-	`id2`	-	Identifier data
12	U4	-	`id3`	-	Identifier data
Start of repeated block (dataSize times)					
16 + 1*N	U1	-	`data`	-	Data for the ALP client
End of repeated block					

30.2.3 ALP client sends AlmanacPlus data to server.

Message	**AID-ALPSRV**				
Description	**ALP client sends AlmanacPlus data to server.**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output Message				
Comment	This message is sent by the ALP client to the ALP server in order to submit updated data. The server can either replace the current data at this position or ignore this new data (which will result in degraded performance).				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x32	8 + 2*size	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`idSize`	bytes	Identifier size
1	U1	-	`type`	-	Set to 0xff to mark that is *not* a data request
2	U2	-	`ofs`	-	Data offset [16bit words]
4	U2	-	`size`	-	Data size [16bit words]
6	U2	-	`fileId`	-	Corresponding ALP file id
Start of repeated block (size times)					
8 + 2*N	U2	-	`data`	-	16bit word data to be submitted to the ALP server
End of repeated block					

30.3 AID-ALP (0x0B 0x50)

30.3.1 ALP file data transfer to the receiver

Message	**AID-ALP**				
Description	**ALP file data transfer to the receiver**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input message				
Comment	This message is used to transfer a chunk of data from the AlmanacPlus file to the receiver. Upon reception of this message, the receiver will write the payload data to its internal non-volatile memory, eventually also erasing that part of the memory first. Make sure that the payload size is even sized (i.e. always a multiple of 2). Do not use payloads larger than ~ 700 bytes, as this would exceed the receiver's internal buffering capabilities. The receiver will (not-) acknowledge this message using the message alternatives given below. The host shall wait for an acknowledge message before sending the next chunk.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x50	0 + 2*N	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
Start of repeated block (N times)					
N*2	U2	-	`alpData`	-	ALP file data
End of repeated block					

30.3.2 Mark end of data transfer

Message	**AID-ALP**				
Description	**Mark end of data transfer**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input message				
Comment	This message is used to indicate that all chunks have been transferred, and normal receiver operation can resume. Upon reception of this message, the receiver will verify all chunks received so far, and enable AssistNow Offline and GPS receiver operation if successful. This message could also be sent to cancel an incomplete download.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x50	1	*see below*	CK_A CK_B
Payload Contents:					

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`dummy`	-	Value is ignored

30.3.3 Acknowledges a data transfer

Message	**AID-ALP**				
Description	**Acknowledges a data transfer**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output message				
Comment	This message from the receiver acknowledges successful processing of a previously received chunk of data with the ``Chunk Transfer`` Message. This message will also be sent once a ``Stop`` message has been received, and the integrity of all chunks received so far has been checked successfully.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x50	1	*see below*	CK_A CK_B
Payload Contents:					

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`ack`	-	Set to 0x01

30.3.4 Indicate problems with a data transfer

Message	**AID-ALP**				
Description	**Indicate problems with a data transfer**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output message				
Comment	This message from the receiver indicates that an error has occurred while processing and storing the data received with the ``Chunk Transfer`` message. This message will also be sent once a stop command has been received, and the integrity of all chunks received failed.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x50	1	*see below*	CK_A CK_B
Payload Contents:					

AID-ALP continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`nak`	-	Set to 0x00

30.3.5 Poll the AlmanacPlus status

Message	**AID-ALP**				
Description	**Poll the AlmanacPlus status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x50	24	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`predTow`	s	Prediction start time of week
4	U4	-	`predDur`	s	Prediction duration from start of first data set to end of last data set
8	I4	-	`age`	s	Current age of ALP data
12	U2	-	`predWno`	-	Prediction start week number
14	U2	-	`almWno`	-	Truncated week number of reference almanac
16	U4	-	`reserved1`	-	Reserved
20	U1	-	`svs`	-	Number of satellite data sets contained in the ALP data
21	U1	-	`reserved2`	-	Reserved
22	U2	-	`reserved3`	-	Reserved

30.4 AID-AOP (0x0B 0x33)

30.4.1 Poll AssistNow Autonomous data

Message	**AID-AOP**				
Description	**Poll AssistNow Autonomous data**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Poll request				
Comment	**This message has an empty payload.** Poll *AssistNow Autonomous* aiding data for all satellits by sending this empty message. The receiver will return an AID-AOP message (see definition below) for each satellite for which data is available. For satellites for which no data is available it will return a corresponding AID-AOP poll request message (see below).				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x33	0	*see below*	CK_A CK_B
No payload					

30.4.2 Poll AssistNow Autonomous data for one satellite

Message	**AID-AOP**				
Description	**Poll AssistNow Autonomous data for one satellite**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Poll request				
Comment	Poll the *AssistNow Autonomous* data for the specified satellite. The receiver will return a AID-AOP message (see definition below) if data is available for the requested satellite. If no data is available it will return corresponding AID-AOP poll request message (i.e. this message).				
Message Structure	*Header*	*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).

30.4.3 AssistNow Autonomous data

Message	**AID-AOP**				
Description	**AssistNow Autonomous data**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Input/Output Message				
Comment	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 AOP data is available or the corresponding poll request message if no AOP 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*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x33	(48) or (192)	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`svid`	-	GPS SV id
1	U1[47]	-	`data`	-	*AssistNow Autonomous* data
Start of optional block					
48	U1[48]	-	`optional0`	-	Optional data chunk 1/3
96	U1[48]	-	`optional1`	-	Optional data chunk 2/3
144	U1[48]	-	`optional2`	-	Optional data chunk 3/3
End of optional block					

30.5 AID-DATA (0x0B 0x10)

30.5.1 Polls all GPS Initial Aiding Data

Message	**AID-DATA**				
Description	**Polls all GPS Initial Aiding Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll				
Comment	If this poll is received, the messages AID-INI, AID-HUI, AID-EPH and AID-ALM are sent.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x10	0	*see below*	CK_A CK_B
No payload					

30.6 AID-EPH (0x0B 0x31)

30.6.1 Poll GPS Aiding Ephemeris Data

Message	**AID-EPH**				
Description	**Poll GPS Aiding Ephemeris Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	**This message has an empty payload!** 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x31	0	*see below*	CK_A CK_B
No payload					

30.6.2 Poll GPS Aiding Ephemeris Data for a SV

Message	**AID-EPH**				
Description	**Poll GPS Aiding Ephemeris Data for a SV**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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).

30.6.3 GPS Aiding Ephemeris Input/Output Message

Message	**AID-EPH**				
Description	**GPS Aiding Ephemeris Input/Output Message**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input/Output Message				
Comment	• 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).				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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					

30.7 AID-HUI (0x0B 0x02)

30.7.1 Poll GPS Health, UTC and ionosphere parameters

Message	**AID-HUI**				
Description	**Poll GPS Health, UTC and ionosphere parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	**This message has an empty payload!** -				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x02	0	*see below*	CK_A CK_B
No payload					

30.7.2 GPS Health, UTC and ionosphere parameters

Message	**AID-HUI**				
Description	**GPS Health, UTC and ionosphere parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input/Output Message				
Comment	This message contains a health bit mask, UTC time and Klobuchar parameters. For more information on these parameters, please see the ICD-GPS-200 documentation.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x02	72	*see below*	CK_A CK_B

Payload Contents:

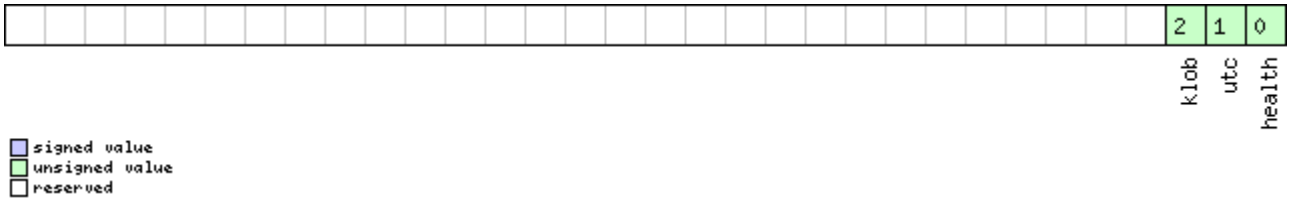
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X4	-	`health`	-	Bitmask, every bit represenst a GPS SV (1-32). If the bit is set the SV is healthy.
4	R8	-	`utcA0`	-	UTC - parameter A0
12	R8	-	`utcA1`	-	UTC - parameter A1
20	I4	-	`utcTOW`	-	UTC - reference time of week
24	I2	-	`utcWNT`	-	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/semicircle	Klobuchar - alpha 1
44	R4	-	`klobA2`	s/semicircle^2	Klobuchar - alpha 2

AID-HUI continued

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

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`health`	Healthmask field in this message is valid
`utc`	UTC parameter fields in this message are valid
`klob`	Klobuchar parameter fields in this message are valid

30.8 AID-INI (0x0B 0x01)

30.8.1 Poll GPS Initial Aiding Data

Message	**AID-INI**				
Description	**Poll GPS Initial Aiding Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	**This message has an empty payload!** -				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x01	0	*see below*	CK_A CK_B
No payload					

30.8.2 Aiding position, time, frequency, clock drift

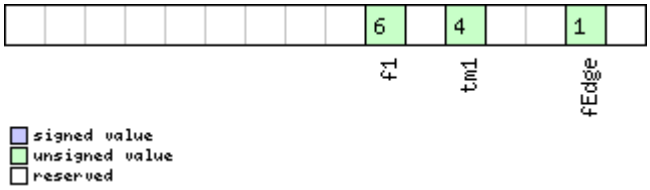
Message	**AID-INI**				
Description	**Aiding position, time, frequency, clock drift**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Polled				
Comment	This message contains position, time and clock drift information. The position can be input in either the 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 baudrate, or using harware time synchronization where an accuracte 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*	*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	-	`wn`	-	Actual week number
20	U4	-	`tow`	ms	Actual time of week
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`

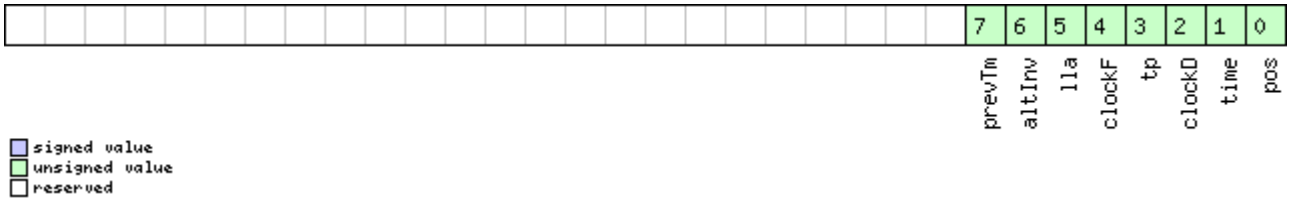


Bitfield tmCfg Description continued

Name	*Description*
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/LON/ALT (default is ECEF)
`altInv`	Altitude is not valid, in case lla was set
`prevTm`	Use time mark received before AID-INI message (default uses mark received after message)

30.9 AID-REQ (0x0B 0x00)

30.9.1 Sends a poll (AID-DATA) for all GPS Aiding Data

Message	**AID-REQ**				
Description	**Sends a poll (AID-DATA) for all GPS Aiding Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Virtual				
Comment	**AID-REQ is not a message but a placeholder for configuration purposes.** If the virtual AID-REQ is configured to be output (see CFG-MSG), the receiver will output a request for aiding data (AID-DATA) after a start-up if its internally stored data (position, time) don't allow it to perform a hot start. If position and time information could be retrieved from internal storage, no AID-REQ will be sent, even when the receiver is missing valid ephemeris data.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0B 0x00	0	*see below*	CK_A CK_B
No payload					

31 CFG (0x06)

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

31.1 CFG-ANT (0x06 0x13)

31.1.1 Poll Antenna Control Settings

Message	**CFG-ANT**				
Description	**Poll Antenna Control Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-ANT with a payload as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x13	0	*see below*	CK_A CK_B
No payload					

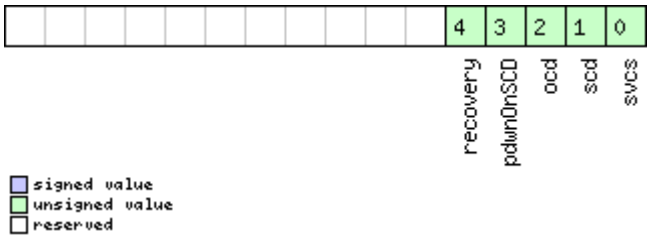
31.1.2 Get/Set Antenna Control Settings

Message	**CFG-ANT**				
Description	**Get/Set Antenna Control Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	-				
Message Structure	*Header*	*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`



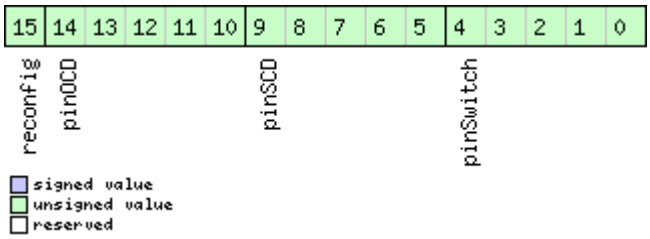
Name	*Description*
`svcs`	Enable Antenna Supply Voltage Control Signal
`scd`	Enable Short Circuit Detection
`ocd`	Enable Open Circuit Detection
`pdwnOnSCD`	Power Down Antenna supply if Short Circuit is detected. (only in combination with Bit 1)

Bitfield flags Description continued

Name	*Description*
`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 (internal to TIM-LP/TIM-LF)
`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.

31.2 CFG-CFG (0x06 0x09)

31.2.1 Clear, Save and Load configurations

Message	**CFG-CFG**				
Description	**Clear, Save and Load configurations**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Command				
Comment	See the Receiver Configuration chapter 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 please refer to the Organization of the Configuration Sections. Please note that commands can be combined. The sequence of execution is Clear, Save, Load				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x09	(12) or (13)	*see below*	CK_A CK_B

Payload Contents:

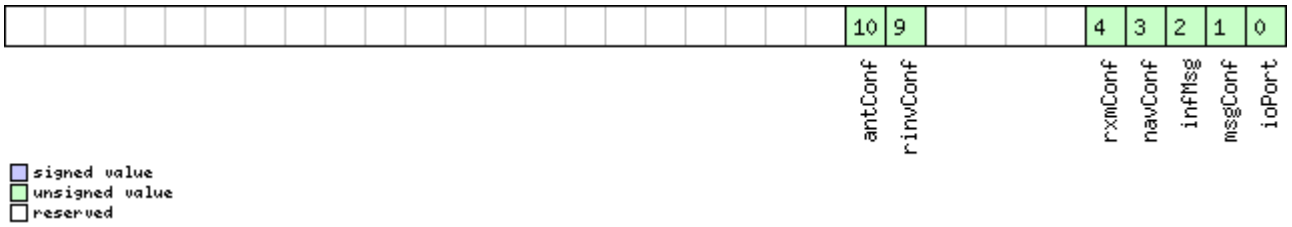
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X4	-	`clearMask`	-	Mask with configuration sub-sections to Clear (=Load Default Configurations to Permanent Configurations in non-volatile memory) (see graphic below)
4	X4	-	`saveMask`	-	Mask with configuration sub-section to Save (=Save Current Configuration to Non-volatile Memory), see ID description of clearMask
8	X4	-	`loadMask`	-	Mask with configuration sub-sections to Load (=Load Permanent Configurations from Non-volatile Memory to Current Configurations), see ID description of clearMask

CFG-CFG continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
Start of optional block					
12	X1	-	`deviceMask`	-	Mask which selects the devices for this command. (see graphic below)
End of optional block					

Bitfield clearMask

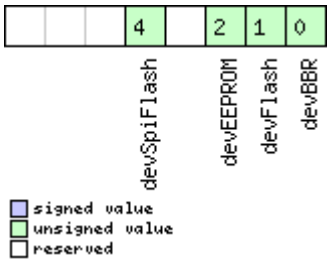
This Graphic explains the bits of `clearMask`



Name	*Description*
`ioPort`	Port Settings
`msgConf`	Message Configuration
`infMsg`	INF Message Configuration
`navConf`	Navigation Configuration
`rxmConf`	Receiver Manager Configuration
`rinvConf`	Remote Inventory Configuration
`antConf`	Antenna Configuration

Bitfield deviceMask

This Graphic explains the bits of `deviceMask`



Name	*Description*
`devBBR`	device battery backed RAM
`devFlash`	device Flash
`devEEPROM`	device EEPROM
`devSpiFlash`	device SPI Flash

31.3 CFG-DAT (0x06 0x06)

31.3.1 Poll Datum Setting

Message	**CFG-DAT**				
Description	**Poll Datum Setting**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Upon sending of this message, the receiver returns CFG-DAT as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x06	0	*see below*	CK_A CK_B
No payload					

31.3.2 Set Standard Datum

Message	**CFG-DAT**				
Description	**Set Standard Datum**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 6.02.				
Type	Set				
Comment	See section Geodetic Datums in the appendix for a list of supported Datums				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x06	2	*see below*	CK_A CK_B
Payload Contents:					

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U2	-	`datumNum`	-	Datum Number

31.3.3 Set User-defined Datum

Message	**CFG-DAT**				
Description	**Set User-defined Datum**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 metres).
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 metres).
20	R4	-	`dY`	m	Y Axis shift at the origin (accepted range is +/- 5000.0 metres).
24	R4	-	`dZ`	m	Z Axis shift at the origin (accepted range is +/- 5000.0 metres).

CFG-DAT continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
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).

31.3.4 Get currently selected Datum

Message	**CFG-DAT**				
Description	**Get currently selected Datum**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get				
Comment	The Parameter datumName is only valid, if datumNum is not equal to -1. In case datumNum is -1, the receiver is configured for a custom datum. The parameters from majA to scale are valid for both custom or standard datum formats.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 according to Geodetic Datums
2	CH[6]	-	`datumName`	-	ASCII String with Datum Mnemonic
8	R8	-	`majA`	m	Semi-major Axis (accepted range = 6,300,000.0 to 6,500,000.0 metres).
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 metres).
28	R4	-	`dY`	m	Y Axis shift at the origin (accepted range is +/- 5000.0 metres).
32	R4	-	`dZ`	m	Z Axis shift at the origin (accepted range is +/- 5000.0 metres).
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).

31.4 CFG-EKF (0x06 0x12)

31.4.1 Poll EKF Module Settings

Message	**CFG-EKF**				
Description	**Poll EKF Module Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with ADR product variant**).				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-EKF with a payload as defined below. This message is only available on EKF (Dead Reckoning) GPS Receivers.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x12	0	*see below*	CK_A CK_B
No payload					

31.4.2 Get/Set EKF Module Settings - LEA-6R

Message	**CFG-EKF**				
Description	**Get/Set EKF Module Settings - LEA-6R**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with ADR product variant**).				
Type	Get/Set				
Comment	This message is only available on EKF (Dead Reckoning) GPS Receivers (LEA-6R).				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x12	16	*see below*	CK_A CK_B
Payload Contents:					

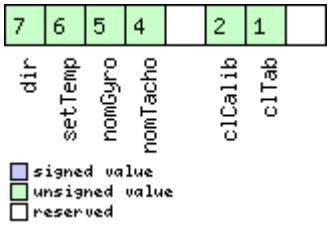
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`disableEkf`	-	1=EKF solution disabled. 0=EKF solution enabled.
1	X1	-	`actionFlags`	-	Meaning in input struct: Config action flags: (see graphic below)
2	U1	-	`configFlags`	-	configuration flags (see graphic below)
3	X1	-	`inverseFlags`	-	The following flags can be used to 'invert' the meaning of the sensor signals (see graphic below)
4	U4	-	`reserved2`	-	Always set to zero
8	U2	-	`nomPPDist`	-	Nominal tacho pulses per distance, permitted range and distance unit (m or km) depend on pulsesPerM flag
10	U2	-	`nomZero`	mV	Nominal gyro zero point output, permitted range: 2000--3000
12	U1	-	`nomSens`	mV/(deg/s)	Nominal gyro sensitivity, permitted range: 20--40
13	U1	0.1	`rmsTemp`	mV	Maximum allowable RMS threshold for zero velocity temperature compensation, permitted range: 1--10

CFG-EKF continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
14	U2	-	`tempUpdate`	s	Temperature table save-to-flash-interval, minimum: 9

Bitfield actionFlags

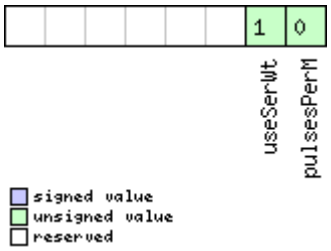
This Graphic explains the bits of `actionFlags`



Name	*Description*
`clTab`	Clear temperature compensation table
`clCalib`	Clear stored calibration
`nomTacho`	Set nominal tacho pulses as indicated below (Field nomPPKM)
`nomGyro`	Set nominal gyro values as indicated below (Fields nomSens, nomZero)
`setTemp`	Set temp table config (Fields tempUpdate, rmsTemp)
`dir`	Set direction pin and gyro sense meaning (inverse_flags)

Bitfield configFlags

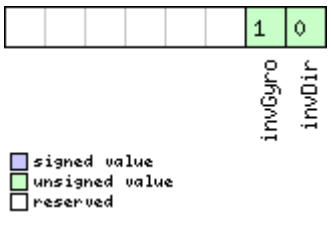
This Graphic explains the bits of `configFlags`



Name	*Description*
`pulsesPerM`	pulses per distance (nomPPDist) is given in pulses per meter 0: field nomPPDist contains pulses per kilometer (permitted range: 1100--45000) 1: field nomPPDist contains pulses per meter
`useSerWt`	use serial wheel tick instead of analog pulse

Bitfield inverseFlags

This Graphic explains the bits of `inverseFlags`



Name	*Description*

Bitfield inverseFlags Description continued

Name	*Description*
`invDir`	invert meaning of direction pin: 0:High=Forwards 1:High=Backwards
`invGyro`	invert meaning of gyro rotation sense: 0:clockwise positive (axis downwards for right handed system), 1:anti-clockwise positive (axis upwards for right handed system)

31.5 CFG-ESFGWT (0x06 0x29)

31.5.1 Get/Set settings of gyro+wheel tick sol (GWT) - LEA-6R

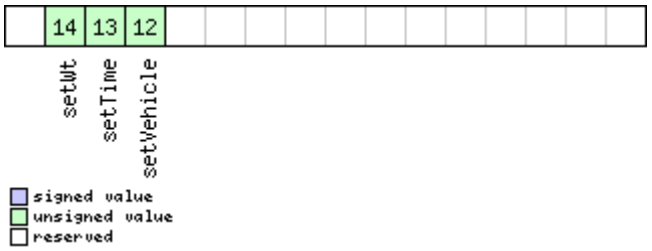
Message	**CFG-ESFGWT**				
Description	**Get/Set settings of gyro+wheel tick sol (GWT) - LEA-6R**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with ADR product variant**).				
Type	Get/Set message				
Comment	Get/Set settings of external gyroscope and wheel tick sensors. A detailed description on how to compose this configuration is given in section Gyro and Wheel Tick (GWT) Solution Configuration (LEA-6R).				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x29	44	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X2	-	`flags`	-	Flags (see graphic below)
2	U2	-	`id`	-	identification number of the sensor data provider, set to zero if not advised differently by u-blox.
4	U4	1e-6	`wtFactor`	-	wheel tick factor to obtain distance [m] from WT (0= not set)
8	U4	-	`reserved1`	-	Reserved
12	U4	1e-6	`wtQuantError`	m	wheel tick quantization error, calculated as 2*PI*wheelRadius divided by ticksPerRotation
16	U4	1e-6	`timeTagFactor`	-	factor of sensor time tag to obtain seconds
20	U4	-	`wtCountMax`	-	maximum value of tick counter (rollover - 1) (0 if no rollover but relative values)
24	U4	-	`timeTagMax`	-	maximum value of sensor time tag (rollover - 1) (0 if no rollover but relative dt values)
28	U2	-	`wtLatency`	ms	latency of wheel tick data due to e. g. CAN-Bus
30	U2	-	`reserved2`	-	Reserved
32	U1	-	`wtFrequency`	Hz	Nominal wheel tick data frequency
33	U1	-	`reserved3`	-	Reserved
34	U2	-	`speedDeadBand`	cm/s	dead band of speed sensor (0 = not set)
36	U4	-	`reserved4`	-	Reserved
40	U4	-	`reserved5`	-	Reserved

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`setVehicle`	apply the vehicle settings
`setTime`	apply the timing settings
`setWt`	apply the wheel tick settings

31.6 CFG-FXN (0x06 0x0E)

31.6.1 Poll FXN configuration

Message	**CFG-FXN**				
Description	**Poll FXN configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Upon sending of this message, the receiver returns CFG-FXN configuration, as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x0E	0	*see below*	CK_A CK_B
No payload					

31.6.2 RXM FixNOW configuration.

Message	**CFG-FXN**				
Description	**RXM FixNOW configuration.**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Command				
Comment	**This message is outdated and supported on u-blox 5/6 only for easier migration from Antaris 4. Please use** CFG-PM2 **instead.** This message only configures the FixNOW Mode, it does not enable it. To enable FixNOW, please use CFG-RXM.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x0E	36	*see below*	CK_A CK_B

Payload Contents:

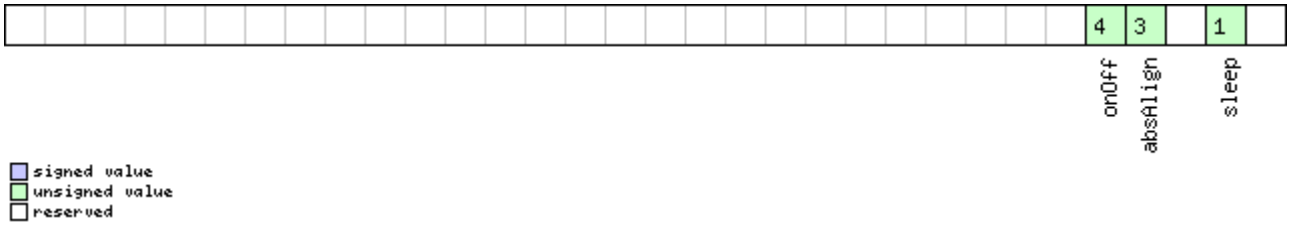
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X4	-	`flags`	-	FXN configuration flags. Bitmask, Combination of the following flags. (see graphic below)
4	U4	-	`tReacq`	ms	Time the receiver tries to re-acquire satellites, before going to off state.
8	U4	-	`tAcq`	ms	Time the receiver tries to acquire satellites, before going to off state.

CFG-FXN continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
12	U4	-	`tReacqOff`	ms	Time the receiver stays in Off-State, if re-acquisition failed.
16	U4	-	`tAcqOff`	ms	Time the receiver stays in Off-State, if acquisition failed.
20	U4	-	`tOn`	ms	On time (starts with first fix)
24	U4	-	`tOff`	ms	Sleep time after normal ontime (actual off time may vary due to data download)
28	U4	-	`reserved1`	-	Reserved
32	U4	-	`baseTow`	ms	Base TOW to which t_on/t_sleep are aligned if ABSOLUTE_ALIGN is set

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`sleep`	If this bit is set, the unit will enter Sleep Mode. Otherwise, it will enter CPU only mode. In Sleep Mode, the RF section and the CPU are shut down. In CPU only Mode, the RF section is shut down, but the CPU continues to run - this mode is suitable for SCK applications, only.
`absAlign`	Absolute Alignment (only with on/off time)
`onOff`	Use on/off time Remaining bits shall never be set.

31.7 CFG-INF (0x06 0x02)

31.7.1 Poll INF message configuration for one protocol

Message	**CFG-INF**				
Description	**Poll INF message configuration for one protocol**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
	Header	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
Message Structure	0xB5 0x62	0x06 0x02	1	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*

CFG-INF continued

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

31.7.2 Information message configuration

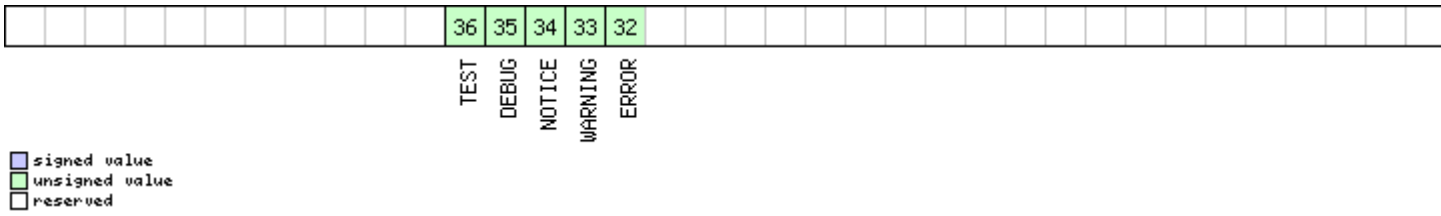
Message	**CFG-INF**				
Description	**Information message configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	The value of INFMSG_mask<x> below are that each bit represents one of the INF class messages (Bit 0 for ERROR, Bit 1 for WARNING and so on.). For a complete list, please 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. Please note that I/O Targets 1 and 2 correspond to serial ports 1 and 2. I/O target 0 is DDC. I/O target 3 is USB. I/O target 4 is SPI. I/O target 5 is reserved for future use.				
Message Structure	*Header*	*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	-	`reserved0`	-	Reserved
2 + 10*N	U2	-	`reserved1`	-	Reserved
4 + 10*N	X1[6]	-	`infMsgMask`	-	A bit mask, saying which information messages are enabled on each I/O target (see graphic below)
End of repeated block					

Bitfield infMsgMask

This Graphic explains the bits of `infMsgMask`



31.8 CFG-ITFM (0x06 0x39)

31.8.1 Jamming/Interference Monitor configuration.

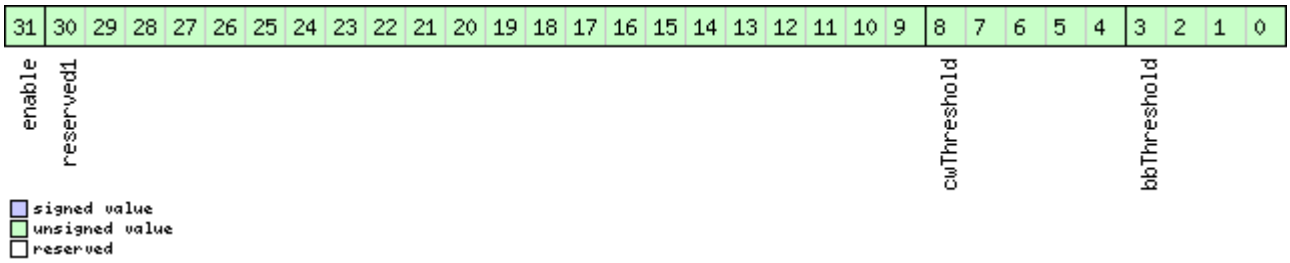
Message	**CFG-ITFM**				
Description	**Jamming/Interference Monitor configuration.**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Command				
Comment	Configuration of Jamming/Interference monitor.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	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

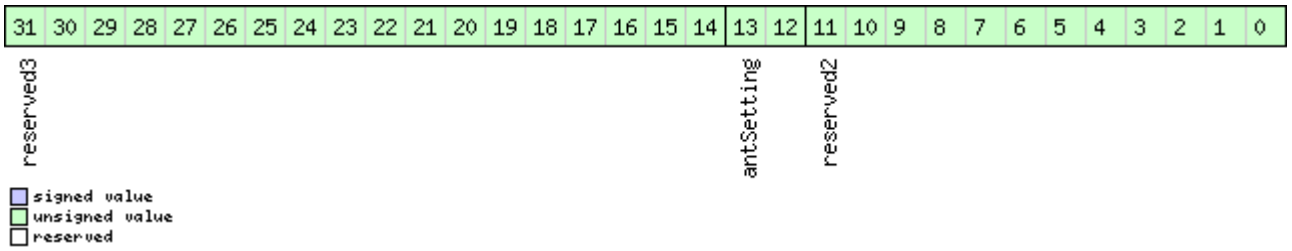
This Graphic explains the bits of `config`



Name	*Description*
`bbThreshold`	Broadband jamming detection threshold (dB)
`cwThreshold`	CW jamming detection threshold (dB)
`reserved1`	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*
`reserved2`	should be set to 0x31E in hex for correct setting
`antSetting`	antennaSetting, 0=unknown, 1=passive, 2=active
`reserved3`	reserved, set to 0

31.9 CFG-MSG (0x06 0x01)

31.9.1 Poll a message configuration

Message	**CFG-MSG**				
Description	**Poll a message configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*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

31.9.2 Set Message Rate(s)

Message	**CFG-MSG**				
Description	**Set Message Rate(s)**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
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*	*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

CFG-MSG continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
2	U1[6]	-	`rate`	-	Send rate on I/O Target (6 Targets)

31.9.3 Set Message Rate

Message	**CFG-MSG**				
Description	**Set Message Rate**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	Set message rate configuration for the current target. See also section How to change between protocols.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Target

31.10 CFG-NAV5 (0x06 0x24)

31.10.1 Poll Navigation Engine Settings

Message	**CFG-NAV5**				
Description	**Poll Navigation Engine Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-NAV5 with a payload as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x24	0	*see below*	CK_A CK_B
No payload					

31.10.2 Get/Set Navigation Engine Settings

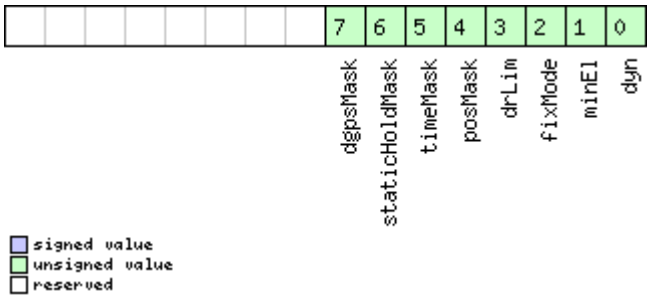
Message	**CFG-NAV5**				
Description	**Get/Set Navigation Engine Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	See the Navigation Configuration Settings Description for a detailed description of how these settings affect receiver operation.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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
3	U1	-	`fixMode`	-	Position Fixing Mode. - 1: 2D only - 2: 3D only - 3: Auto 2D/3D
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	Maximum time to perform dead reckoning (linear extrapolation) in case of GPS signal loss
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	-	`dgpsTimeOut`	s	DGPS timeout, firmware 7 and newer only
24	U4	-	`reserved2`	-	Always set to zero
28	U4	-	`reserved3`	-	Always set to zero
32	U4	-	`reserved4`	-	Always set to zero

Bitfield mask

This Graphic explains the bits of `mask`



Name	Description
`dyn`	Apply dynamic model settings
`minEl`	Apply minimum elevation settings
`fixMode`	Apply fix mode settings
`drLim`	Apply DR limit settings
`posMask`	Apply position mask settings
`timeMask`	Apply time mask settings
`staticHoldMask`	Apply static hold settings
`dgpsMask`	Apply DGPS settings, firmware 7 and newer only

31.11 CFG-NAVX5 (0x06 0x23)

31.11.1 Poll Navigation Engine Expert Settings

Message	**CFG-NAVX5**				
Description	**Poll Navigation Engine Expert Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-NAVX5 with a payload as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x23	0	*see below*	CK_A CK_B
No payload					

31.11.2 Get/Set Navigation Engine Expert Settings

Message	**CFG-NAVX5**				
Description	**Get/Set Navigation Engine Expert Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x23	40	*see below*	CK_A CK_B
Payload Contents:					

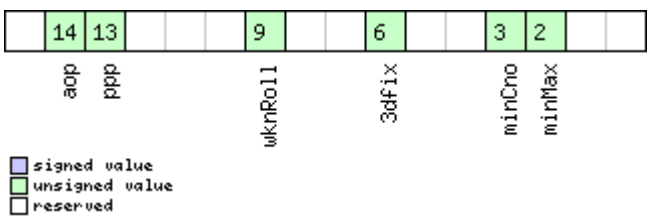
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U2	-	`version`	-	Message version. Current version is 0.

CFG-NAVX5 continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
2	X2	-	`mask1`	-	First Parameters Bitmask. Only the flagged parameters will be applied, unused bits must be set to 0. (see graphic below)
4	U4	-	`reserved0`	-	Always set to zero
8	U1	-	`reserved1`	-	Always set to zero
9	U1	-	`reserved2`	-	Always set to zero
10	U1	-	`minSVs`	#SVs	Minimum number of satellites for navigation
11	U1	-	`maxSVs`	#SVs	Maximum number of satellites for navigation
12	U1	-	`minCNO`	dbHz	Minimum satellite signal level for navigation
13	U1	-	`reserved5`	-	Always set to zero
14	U1	-	`iniFix3D`	-	Initial Fix must be 3D flag (0=false/1=true)
15	U1	-	`reserved6`	-	Always set to zero
16	U1	-	`reserved7`	-	Always set to zero
17	U1	-	`reserved8`	-	Always set to zero
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	U4	-	`reserved9`	-	Always set to zero
24	U1	-	`reserved10`	-	Always set to zero
25	U1	-	`reserved11`	-	Always set to zero
26	U1	-	`usePPP`	-	use Precise Point Positioning flag (0=false/1=true)
27	U1	-	`useAOP`	-	*AssistNow Autonomous*, see the receiver description for details on this feature - 1 = enabled - 0 = disabled (default)
28	U1	-	`reserved12`	-	Always set to zero
29	U1	-	`reserved13`	-	Always set to zero
30	U2	-	`aopOrbMaxErr`	[m]	maximum acceptable (modelled) *AssistNow Autonomous* orbit error (valid range = 5..1000, or 0 = reset to firmware default)
32	U4	-	`reserved3`	-	Always set to zero
36	U4	-	`reserved4`	-	Always set to zero

Bitfield mask1

This Graphic explains the bits of `mask1`



Name	*Description*
`minMax`	Apply min/max SVs settings

Bitfield mask1 Description continued

Name	*Description*
`minCno`	Apply minimum C/N0 setting
`3dfix`	Apply initial 3D fix settings
`wknRoll`	Apply GPS weeknumber rollover settings
`ppp`	Apply PPP flag (see PPP configuration)
`aop`	Apply useAOP flag and aopOrbMaxErr setting (AssistNow Autonomous)

31.12 CFG-NMEA (0x06 0x17)

31.12.1 Poll the NMEA protocol configuration

Message	**CFG-NMEA**				
Description	**Poll the NMEA protocol configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x17	0	*see below*	CK_A CK_B
No payload					

31.12.2 Set/Get the NMEA protocol configuration

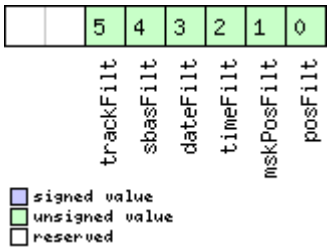
Message	**CFG-NMEA**				
Description	**Set/Get the NMEA protocol configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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	-	`version`	-	0x23 = NMEA version 2.3 0x21 = NMEA version 2.1
2	U1	-	`numSV`	-	Maximum Number of SVs to report in NMEA protocol. This does not affect the receiver's operation. It only limits the number of SVs reported in NMEA mode (this might be needed with older mapping applications which only support 8- or 12-channel receivers).
3	X1	-	`flags`	-	flags (see graphic below)

Bitfield filter

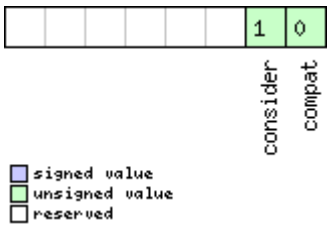
This Graphic explains the bits of `filter`



Name	*Description*
`posFilt`	disable position filtering
`mskPosFilt`	disable masked position filtering
`timeFilt`	disable time filtering
`dateFilt`	disable date filtering
`sbasFilt`	enable SBAS filtering
`trackFilt`	disable track filtering

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.

31.13 CFG-NVS (0x06 0x22)

31.13.1 Clear, Save and Load non-volatile storage data

Message	**CFG-NVS**				
Description	**Clear, Save and Load non-volatile storage data**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Command				
Comment	Three masks are made up of individual bits that indicate which data is to be cleared, saved and/or loaded. The fourth mask defines on which devices the corresponding action shall be carried out. Please note that only one command should be flagged at once. Otherwise all commands are processed in the order Clear, Save, and Load. All reserved bits must be set to zero.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x22	13	*see below*	CK_A CK_B

Payload Contents:

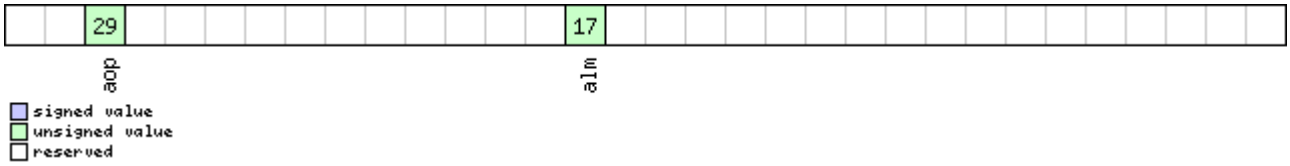
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*

CFG-NVS continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X4	-	`clearMask`	-	Mask of data to be cleared (see graphic below)
4	X4	-	`saveMask`	-	Mask of data to be saved, uses the same bits as the clearMask
8	X4	-	`loadMask`	-	Mask of data to be loaded, uses the same bits as the clearMask
12	X1	-	`deviceMask`	-	Mask of devices to consider (default: all devices) (see graphic below)

Bitfield clearMask

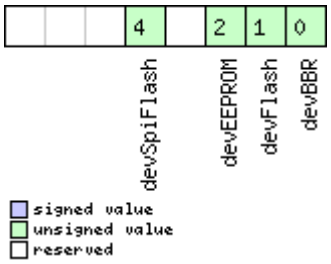
This Graphic explains the bits of `clearMask`



Name	*Description*
`alm`	GPS Almanac data
`aop`	AOP data

Bitfield deviceMask

This Graphic explains the bits of `deviceMask`



Name	*Description*
`devBBR`	built-in battery-backed RAM
`devFlash`	external flash memory
`devEEPROM`	external EEPROM
`devSpiFlash`	external SPI Flash (only U5R6 and later)

31.14 CFG-PM2 (0x06 0x3B)

31.14.1 Poll extended Power Management configuration

Message	**CFG-PM2**				
Description	**Poll extended Power Management configuration**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x3B	0	*see below*	CK_A CK_B
No payload					

31.14.2 Extended Power Management configuration

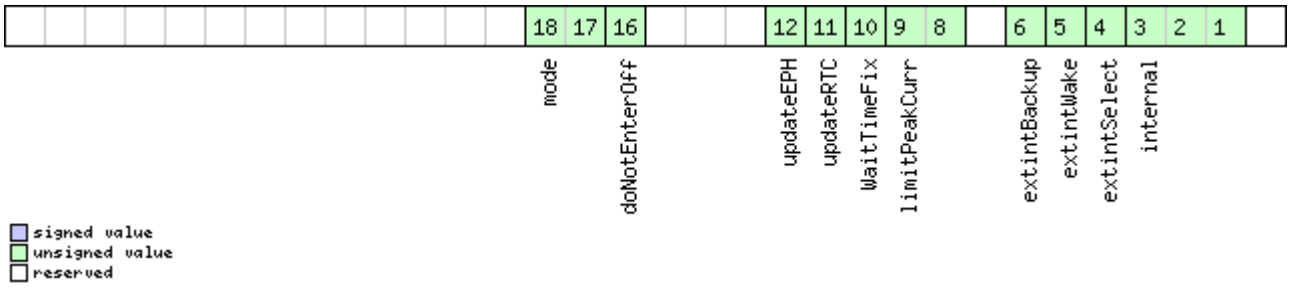
Message	**CFG-PM2**				
Description	**Extended Power Management configuration**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Set/Get				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x3B	44	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`version`	-	Message version (set to 1)
1	U1	-	`reserved1`	-	Reserved
2	U1	-	`reserved2`	-	Reserved
3	U1	-	`reserved3`	-	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. For possible restrictions see Restrictions.
12	U4	-	`searchPeriod`	ms	Acquisition retry period. If set to 0, the receiver will never retry a startup
16	U4	-	`gridOffset`	ms	Grid offset relative to GPS start of week
20	U2	-	`onTime`	s	on time after first successful fix
22	U2	-	`minAcqTime`	s	minimal search time
24	U2	-	`reserved4`	-	Reserved
26	U2	-	`reserved5`	-	Reserved
28	U4	-	`reserved6`	-	Reserved
32	U4	-	`reserved7`	-	Reserved
36	U1	-	`reserved8`	-	Reserved
37	U1	-	`reserved9`	-	Reserved
38	U2	-	`reserved10`	-	Reserved
40	U4	-	`reserved11`	-	Reserved

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`internal`	Internal Flag: **Must be set to '000'**
`extintSelect`	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'
`extintBackup`	EXTINT Pin Control 0 disabled 1 enabled, force receiver into BACKUP mode when selected EXTINT pin is 'low'
`limitPeakCurr`	Limit Peak Current 00 disabled 01 enabled, peak current is limited 10 reserved 11 reserved
`WaitTimeFix`	Wait for Timefix 0 wait for normal Fix ok, before starting on-time 1 wait for time fix ok, before starting on-time
`updateRTC`	Update Real Time Clock 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.
`updateEPH`	Update Ephemeris 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 0 receiver enters *inactive for search* state 1 receiver does not enter *inactive for search* state but keeps trying to acquire a fix instead
`mode`	Mode of operation 00 ON/OFF operation 01 Cyclic tracking operation 10 reserved 11 reserved

31.15 CFG-PM (0x06 0x32)

31.15.1 Poll Power Management configuration

Message	**CFG-PM**				
Description	**Poll Power Management configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x32	0	*see below*	CK_A CK_B
No payload					

31.15.2 Power Management configuration

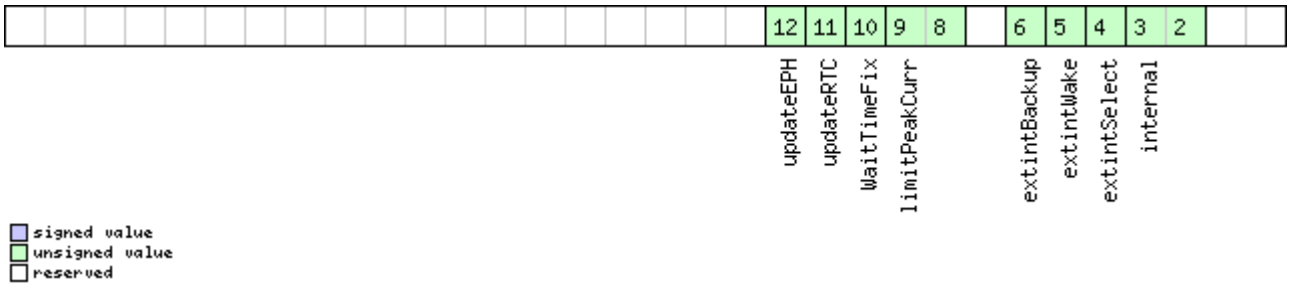
Message	**CFG-PM**				
Description	**Power Management configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	This message is outdated and provided for backward compatibility only. Please use the message UBX-CFG-PM2 instead.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x32	24	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`version`	-	Message version (set to 0)
1	U1	-	`reserved1`	-	Reserved
2	U1	-	`reserved2`	-	Reserved
3	U1	-	`reserved3`	-	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. For possible restrictions see Restrictions.
12	U4	-	`searchPeriod`	ms	Acquisition retry period. If set to 0, the receiver will never retry a startup
16	U4	-	`gridOffset`	ms	Grid offset relative to GPS start of week
20	U2	-	`onTime`	s	on time after first successful fix
22	U2	-	`minAcqTime`	s	minimal search time

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`internal`	Internal Flag: **Must be set to '01'**
`extintSelect`	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'
`extintBackup`	EXTINT Pin Control 0 disabled 1 enabled, force receiver into BACKUP mode when selected EXTINT pin is 'low'
`limitPeakCurr`	Limit Peak Current 00 disabled 01 enabled, peak current is limited 10 reserved 11 reserved
`WaitTimeFix`	Wait for Timefix 0 wait for normal Fix ok, before starting on-time 1 wait for time fix ok, before starting on-time
`updateRTC`	Update Real Time Clock 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.
`updateEPH`	Update Ephemeris 0 Do not wake-up to update Ephemeris data 1 Update Ephemeris. The receiver adds extra wake-up cycles to update the Ephemeris data

31.16 CFG-PRT (0x06 0x00)

31.16.1 Polls the configuration of the used I/O Port

Message	**CFG-PRT**				
Description	**Polls the configuration of the used I/O Port**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Polls the configuration of the I/O Port on which this message is received				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x00	0	*see below*	CK_A CK_B
No payload					

31.16.2 Polls the configuration for one I/O Port

Message	**CFG-PRT**				
Description	**Polls the configuration for one I/O Port**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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*	*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)

31.16.3 Get/Set Port Configuration for UART

Message	**CFG-PRT**				
Description	**Get/Set Port Configuration for UART**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x00	20	*see below*	CK_A CK_B

Payload Contents:

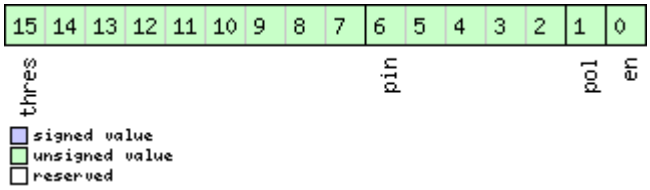
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`portID`	-	Port Identifier Number (= 1 or 2 for UART ports)
1	U1	-	`reserved0`	-	Reserved
2	X2	-	`txReady`	-	reserved (Alwyas set to zero) up to Firmware 7.01, TX ready PIN configuration (since Firmware 7.01) (see graphic below)

CFG-PRT continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
4	X4	-	`mode`	-	A bit mask describing the UART mode (see graphic below)
8	U4	-	`baudRate`	Bits/s	Baudrate in bits/second
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	U2	-	`reserved4`	-	Always set to zero
18	U2	-	`reserved5`	-	Always set to zero

Bitfield txReady

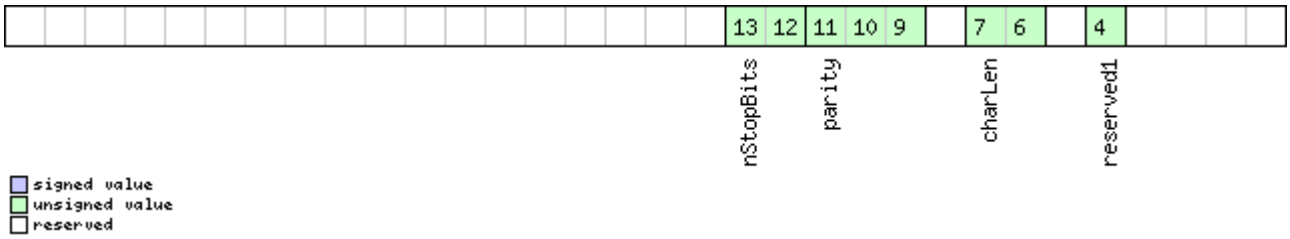
This Graphic explains the bits of `txReady`



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

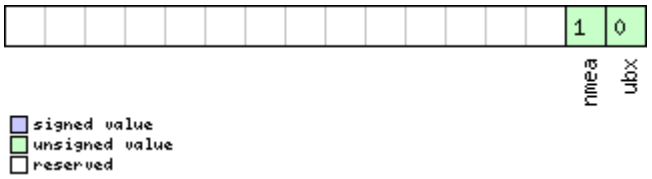
This Graphic explains the bits of `mode`



Name	*Description*
`reserved1`	Default 1 for compatibility with A4
`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

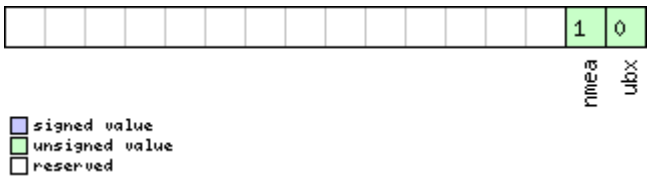
Bitfield inProtoMask

This Graphic explains the bits of `inProtoMask`



Bitfield outProtoMask

This Graphic explains the bits of `outProtoMask`



31.16.4 Get/Set Port Configuration for USB Port

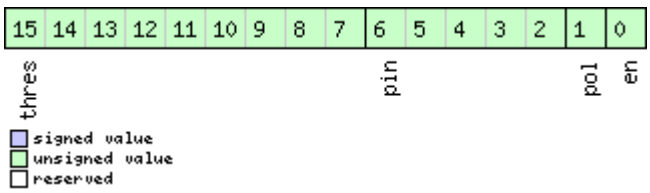
Message	**CFG-PRT**				
Description	**Get/Set Port Configuration for USB Port**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit.				
Message Structure	*Header*	*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 (= 3 for USB port)
1	U1	-	`reserved0`	-	Reserved
2	X2	-	`txReady`	-	reserved (Always set to zero) up to Firmware 7.01, TX ready PIN configuration (since Firmware 7.01) (see graphic below)
4	U4	-	`reserved2`	-	Reserved
8	U4	-	`reserved3`	-	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	U2	-	`reserved4`	-	Always set to zero
18	U2	-	`reserved5`	-	Always set to zero

Bitfield txReady

This Graphic explains the bits of `txReady`



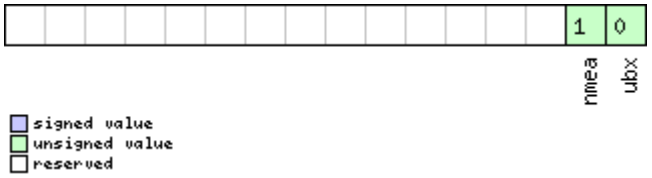
Name	*Description*
`en`	Enable TX ready feature for this port
`pol`	Polarity 0 High-active 1 Low-active

Bitfield txReady Description continued

Name	*Description*
`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

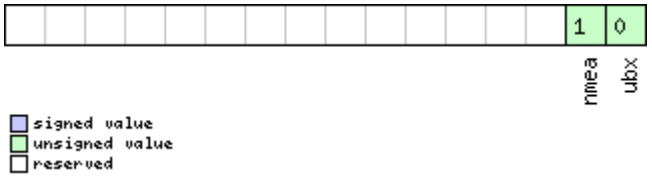
Bitfield inProtoMask

This Graphic explains the bits of `inProtoMask`



Bitfield outProtoMask

This Graphic explains the bits of `outProtoMask`



31.16.5 Get/Set Port Configuration for SPI Port

Message	**CFG-PRT**				
Description	**Get/Set Port Configuration for SPI Port**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x00	20	*see below*	CK_A CK_B

Payload Contents:

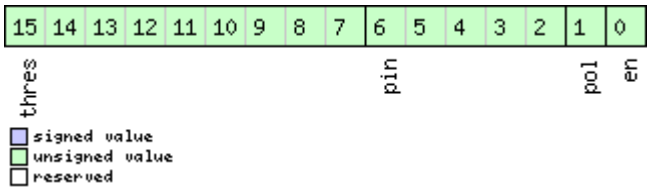
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`portID`	-	Port Identifier Number (= 4 for SPI port)
1	U1	-	`reserved0`	-	Reserved

CFG-PRT continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
2	X2	-	`txReady`	-	reserved (set to 0) up to Firmware 7.01, TX ready PIN configuration (since Firmware 7.01) (see graphic below)
4	X4	-	`mode`	-	SPI Mode Flags (see graphic below)
8	U4	-	`reserved3`	-	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	U2	-	`reserved4`	-	Always set to zero
18	U2	-	`reserved5`	-	Always set to zero

Bitfield txReady

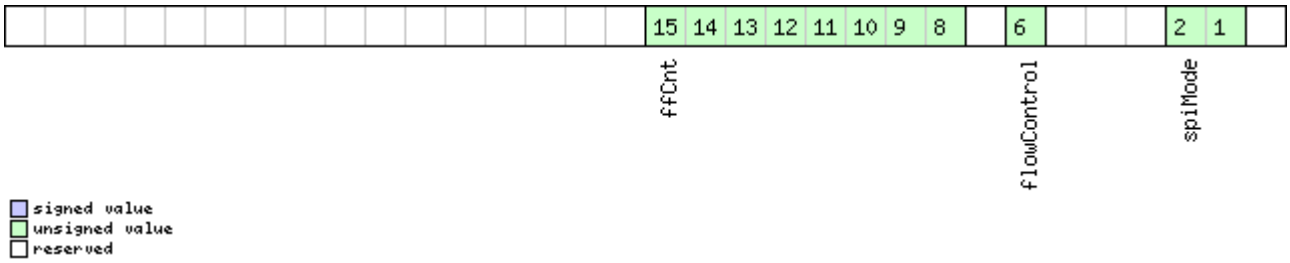
This Graphic explains the bits of `txReady`



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

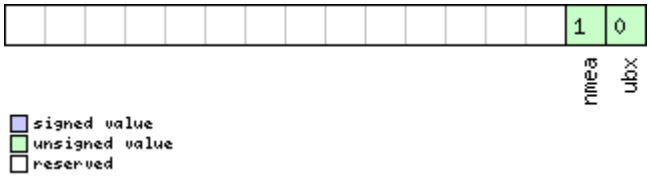
This Graphic explains the bits of `mode`



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
`flowControl`	(u-blox 6 only) 0 Flow control disabled 1 Flow control enabled (9-bit mode)
`ffCnt`	Number of bytes containing 0xFF to receive before switching off reception. Range: 0(mechanism off)-255

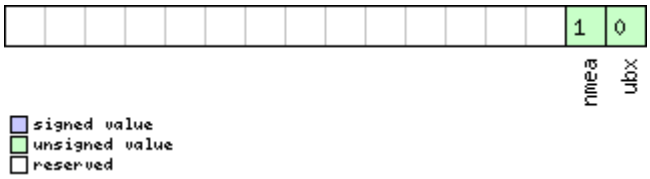
Bitfield inProtoMask

This Graphic explains the bits of `inProtoMask`



Bitfield outProtoMask

This Graphic explains the bits of `outProtoMask`



31.16.6 Get/Set Port Configuration for DDC Port

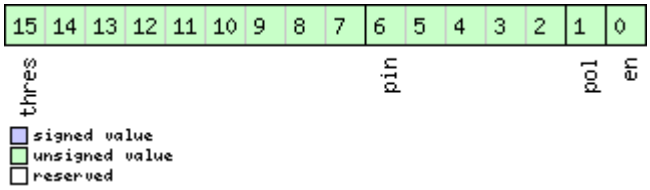
Message	**CFG-PRT**				
Description	**Get/Set Port Configuration for DDC Port**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit.				
Message Structure	*Header*	*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 (= 0 for DDC port)
1	U1	-	`reserved0`	-	Reserved
2	X2	-	`txReady`	-	reserved (set to 0) up to Firmware 7.01, TX ready PIN configuration (since Firmware 7.01) (see graphic below)
4	X4	-	`mode`	-	DDC Mode Flags (see graphic below)
8	U4	-	`reserved3`	-	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	U2	-	`reserved4`	-	Always set to zero
18	U2	-	`reserved5`	-	Always set to zero

Bitfield txReady

This Graphic explains the bits of `txReady`



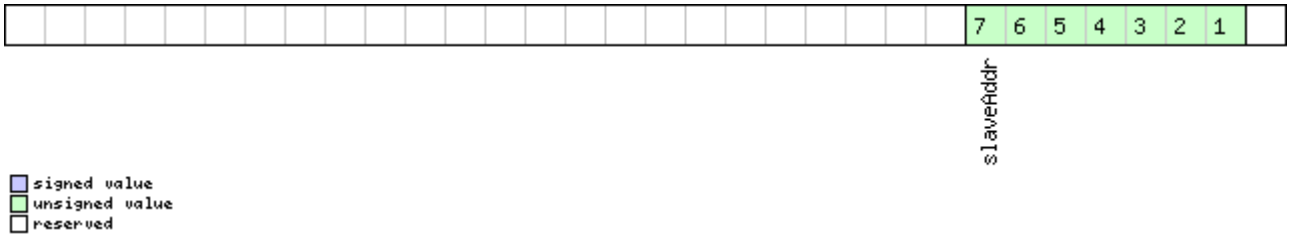
Name	*Description*
`en`	Enable TX ready feature for this port
`pol`	Polarity 0 High-active 1 Low-active
`pin`	PIO to be used (must not be in use already by another function)

Bitfield txReady Description continued

Name	*Description*
`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

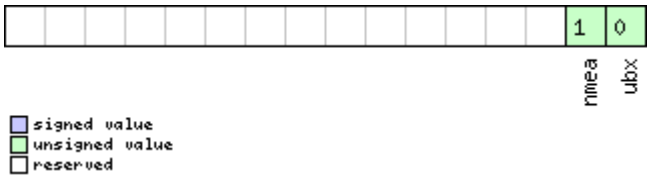
This Graphic explains the bits of `mode`



Name	*Description*
`slaveAddr`	Slave address Range: 0x07 < slaveAddr < 0x78. Bit 0 must be 0

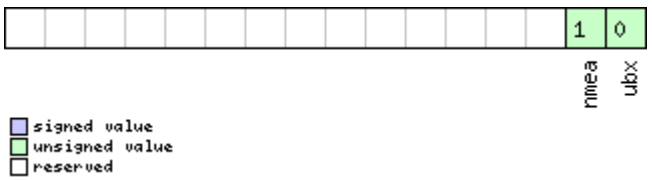
Bitfield inProtoMask

This Graphic explains the bits of `inProtoMask`



Bitfield outProtoMask

This Graphic explains the bits of `outProtoMask`



31.17 CFG-RATE (0x06 0x08)

31.17.1 Poll Navigation/Measurement Rate Settings

Message	**CFG-RATE**				
Description	**Poll Navigation/Measurement Rate Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-RATE with a payload as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x08	0	*see below*	CK_A CK_B
No payload					

31.17.2 Navigation/Measurement Rate Settings

Message	**CFG-RATE**				
Description	**Navigation/Measurement Rate Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	The u-blox positioning technology supports navigation update rates higher or lower than 1 update per second. The calculation of the navigation solution will always be aligned to the top of a second. • 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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	Measurement Rate, GPS measurements are taken every measRate milliseconds
2	U2	-	`navRate`	cycles	Navigation Rate, in number of measurement cycles. On u-blox 5 and u-blox 6, this parameter cannot be changed, and is always equals 1.
4	U2	-	`timeRef`	-	Alignment to reference time: 0 = UTC time, 1 = GPS time

31.18 CFG-RINV (0x06 0x34)

31.18.1 Poll contents of Remote Inventory

Message	**CFG-RINV**				
Description	**Poll contents of Remote Inventory**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x34	0	*see below*	CK_A CK_B
No payload					

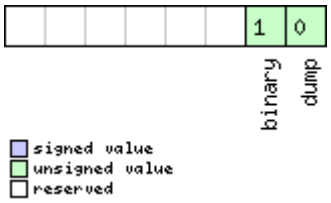
31.18.2 Set/Get contents of Remote Inventory

Message	**CFG-RINV**				
Description	**Set/Get contents of Remote Inventory**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	If *N* is greater than 30, the excess bytes are discarded. In future firmware versions, this limit may change.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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

31.19 CFG-RST (0x06 0x04)

31.19.1 Reset Receiver / Clear Backup Data Structures

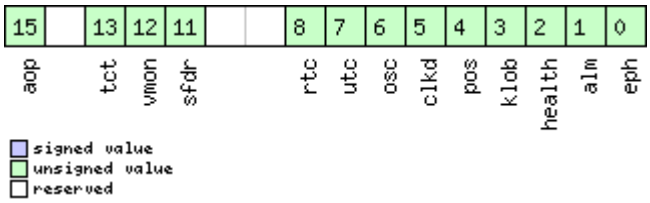
Message	**CFG-RST**				
Description	**Reset Receiver / Clear Backup Data Structures**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Command				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x04	4	*see 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 Hotstart 0x0001 Warmstart 0xFFFF Coldstart (see graphic below)
2	U1	-	`resetMode`	-	Reset Type - 0x00 - Hardware reset (Watchdog) immediately - 0x01 - Controlled Software reset - 0x02 - Controlled Software reset (GPS only) - 0x04 - Hardware reset (Watchdog) after shutdown (>=FW6.0) - 0x08 - Controlled GPS stop - 0x09 - Controlled GPS start
3	U1	-	`reserved1`	-	Reserved

Bitfield navBbrMask

This Graphic explains the bits of `navBbrMask`



Name	*Description*
`eph`	Ephemeris
`alm`	Almanach
`health`	Health
`klob`	Klobuchard
`pos`	Position
`clkd`	Clock Drift
`osc`	Oscilator Parameter
`utc`	UTC Correction Parameters
`rtc`	RTC
`sfdr`	SFDR Parameters

Bitfield navBbrMask Description continued

Name	*Description*
vmon	SFDR Vehicle Monitoring Parameters
tct	TCT Parameters
aop	Autonomous Orbit Parameters

31.20 CFG-RXM (0x06 0x11)

31.20.1 Poll RXM configuration

Message	**CFG-RXM**				
Description	**Poll RXM configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Upon sending of this message, the receiver returns CFG-RXM as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x11	0	*see below*	CK_A CK_B
No payload					

31.20.2 RXM configuration

Message	**CFG-RXM**				
Description	**RXM configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Set/Get				
Comment	For a detailed description see section Power Management.				
Message Structure	*Header*	*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	-	Always set to 8
1	U1	-	lpMode	-	Low Power Mode 0: Max. performance mode 1: Power Save Mode (>= FW 6.00 only) 2-3: reserved 4: Eco mode 5-255: reserved

31.21 CFG-SBAS (0x06 0x16)

31.21.1 Poll contents of SBAS Configuration

Message	**CFG-SBAS**				
Description	**Poll contents of SBAS Configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x16	0	*see below*	CK_A CK_B
No payload					

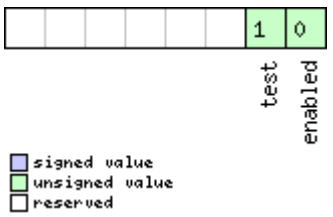
31.21.2 SBAS Configuration

Message	**CFG-SBAS**				
Description	**SBAS Configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Command				
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*	*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)
2	U1	-	`maxSBAS`	-	Maximum Number of SBAS prioritized tracking channels (valid range: 0 - 3) to use
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`

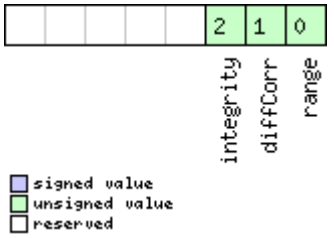


Bitfield mode Description continued

Name	*Description*
Name	*Description*
`enabled`	SBAS Enabled (1) / Disabled (0)
`test`	SBAS Testbed: Use data anyhow (1) / Ignore data when in Test Mode (SBAS Msg 0)

Bitfield usage

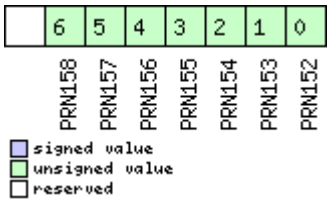
This Graphic explains the bits of `usage`



Name	*Description*
`range`	Use SBAS GEOs as a ranging source (for navigation)
`diffCorr`	Use SBAS Differential Corrections
`integrity`	Use SBAS Integrity Information

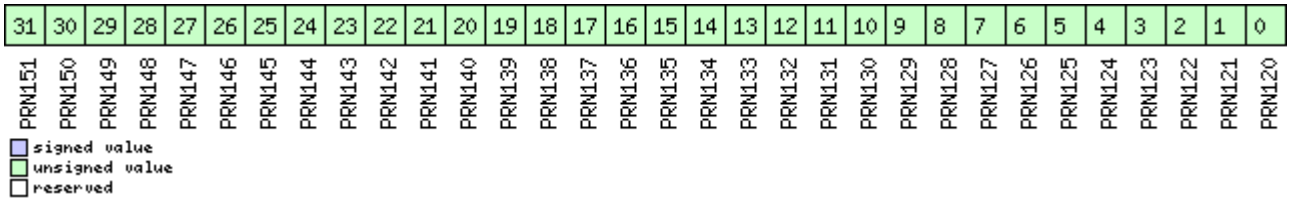
Bitfield scanmode2

This Graphic explains the bits of `scanmode2`



Bitfield scanmode1

This Graphic explains the bits of `scanmode1`



31.22 CFG-TMODE2 (0x06 0x3D)

31.22.1 Poll Time Mode Settings

Message	**CFG-TMODE2**				
Description	**Poll Time Mode Settings**				
Firmware	Supported on u-blox 6 firmware version 7.03 (**only available with timing product variant**).				
Type	Poll Request				
Comment	**This message is available only for timing receivers** Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-TMODE2 with a payload as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x3D	0	*see below*	CK_A CK_B
No payload					

31.22.2 Time Mode Settings 2

Message	**CFG-TMODE2**				
Description	**Time Mode Settings 2**				
Firmware	Supported on u-blox 6 firmware version 7.03 (**only available with timing product variant**).				
Type	Get/Set				
Comment	**This message is available only for timing receivers** See the Time Mode Description for details. This message replaces the deprecated `UBX-CFG-TMODE` message.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x3D	28	*see below*	CK_A CK_B

Payload Contents:

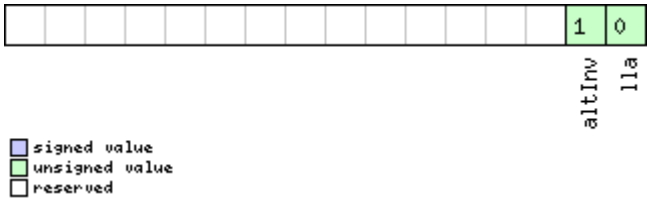
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`timeMode`	-	Time Transfer Mode: 0 Disabled 1 Survey In 2 Fixed Mode (true position information required) 3-255 Reserved
1	U1	-	`reserved1`	-	Reserved
2	X2	-	`flags`	-	Time mode flags (see graphic below)
4	I4	-	`ecefXOrLat`	cm_or_deg*1e-7	WGS84 ECEF X coordinate or latitude, depending on flags above
8	I4	-	`ecefYOrLon`	cm_or_deg*1e-7	WGS84 ECEF Y coordinate or longitude, depending on flags above
12	I4	-	`ecefZOrAlt`	cm	WGS84 ECEF Z coordinate or altitude, depending on flags above
16	U4	-	`fixedPosAcc`	mm	Fixed position 3D accuracy
20	U4	-	`svinMinDur`	s	Survey-in minimum duration

CFG-TMODE2 continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
24	U4	-	`svinAccLimit`	mm	Survey-in position accuracy limit

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`lla`	Position is given in LAT/LON/ALT (default is ECEF)
`altInv`	Altitude is not valid, in case lla was set

31.23 CFG-TMODE (0x06 0x1D)

31.23.1 Poll Time Mode Settings

Message	**CFG-TMODE**				
Description	**Poll Time Mode Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with timing product variant**).				
Type	Poll Request				
Comment	**This message is available only for timing receivers** Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-TMODE with a payload as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x1D	0	*see below*	CK_A CK_B
No payload					

31.23.2 Time Mode Settings

Message	**CFG-TMODE**				
Description	**Time Mode Settings**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with timing product variant**).				
Type	Get/Set				
Comment	**This message is available only for timing receivers. The use of this message is deprecated, starting with firmware version 7.0 please use CFG-TMODE2.** See the Time Mode Description for details.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x1D	28	*see below*	CK_A CK_B
Payload Contents:					

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*

CFG-TMODE continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`timeMode`	-	Time Transfer Mode: 0 Disabled 1 Survey In 2 Fixed Mode (true position information required) 3-255 Reserved
4	I4	-	`fixedPosX`	cm	Fixed Position ECEF X coordinate
8	I4	-	`fixedPosY`	cm	Fixed Position ECEF Y coordinate
12	I4	-	`fixedPosZ`	cm	Fixed Position ECEF Z coordinate
16	U4	-	`fixedPosVar`	mm^2	Fixed position 3D variance
20	U4	-	`svinMinDur`	s	Survey-in minimum duration
24	U4	-	`svinVarLimit`	mm^2	Survey-in position variance limit

31.24 CFG-TP5 (0x06 0x31)

31.24.1 Poll Timepulse Parameters

Message	**CFG-TP5**				
Description	**Poll Timepulse Parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-TP5 with a payload as defined below for Timepulse 0				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x31	0	*see below*	CK_A CK_B
No payload					

31.24.2 Poll TimePulse Parameters

Message	**CFG-TP5**				
Description	**Poll TimePulse Parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this message to the receiver results in the receiver returning a message of type CFG-TP5 with a payload as defined below for the specified Timepulse				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x31	1	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`tpIdx`	-	Timepulse selection (0 = TIMEPULSE, 1 = TIMEPULSE2)

31.24.3 Get/Set TimePulse Parameters

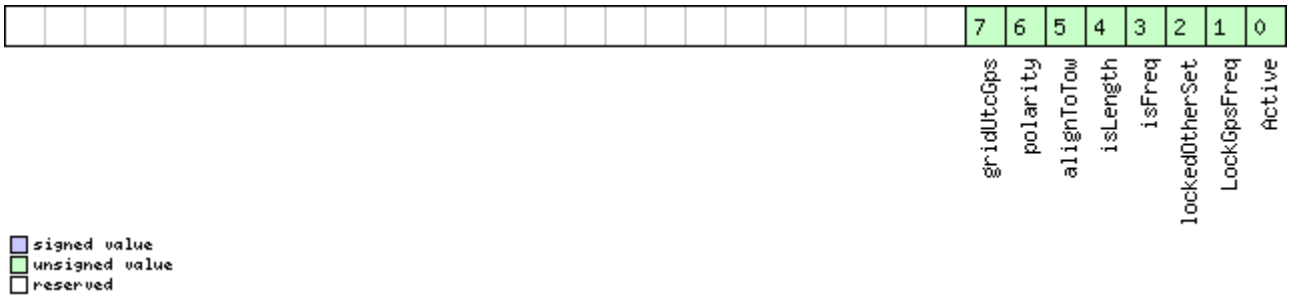
Message	**CFG-TP5**				
Description	**Get/Set TimePulse Parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x31	32	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`tpIdx`	-	Timepulse selection (0 = TIMEPULSE, 1 = TIMEPULSE2)
1	U1	-	`reserved0`	-	Reserved
2	U2	-	`reserved1`	-	Reserved
4	I2	-	`antCableDelay`	ns	Antenna cable delay
6	I2	-	`rfGroupDelay`	ns	RF group delay
8	U4	-	`freqPeriod`	Hz/us	Frequency or period time, depending on setting of bit 'isFreq'
12	U4	-	`freqPeriodLock`	Hz/us	Frequency or period time when locked to GPS time, only used if 'lockedOtherSet' is set
16	U4	1/2^-32	`pulseLenRatio`	us/-	Pulse length or duty cycle, depending on 'isLength'
20	U4	1/2^-32	`pulseLenRatioLock`	us/-	Pulse length or duty cycle when locked to GPS time, only used if 'lockedOtherSet' is set
24	I4	-	`userConfigDelay`	ns	User configurable timepulse delay
28	X4	-	`flags`	-	Configuration flags (see graphic below)

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`Active`	if set enable timepulse; if pin assigned to another function, other function takes precedence
`LockGpsFreq`	if set synchronize Timepulse to GPS as soon as GPS time is valid, otherwise use local clock
`lockedOtherSet`	if set use 'freqPeriodLock' and 'pulseLenRatioLock' as soon as GPS time is valid and 'freqPeriod' and 'pulseLenRatio' if GPS time is invalid, if flag is cleared 'freqPeriod' and 'pulseLenRatio' used regardless of GPS time
`isFreq`	if set 'freqPeriodLock' and 'freqPeriod' interpreted as frequency , otherwise interpreted as period

Bitfield flags Description continued

Name	*Description*
`isLength`	if set 'pulseLenRatioLock' and 'pulseLenRatio' interpreted as pulselength , otherwise interpreted as duty cycle
`alignToTow`	align pulse to top of second (period time must be integer fraction of 1s)
`polarity`	pulse polarity: 0=falling edge at top of second, 1=rising edge at top of second
`gridUtcGps`	timegrid to use: 0=UTC, 1=GPS

31.25 CFG-TP (0x06 0x07)

31.25.1 Poll TimePulse Parameters

Message	**CFG-TP**				
Description	**Poll TimePulse Parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-TP with a payload as defined below				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x07	0	*see below*	CK_A CK_B
No payload					

31.25.2 Get/Set TimePulse Parameters

Message	**CFG-TP**				
Description	**Get/Set TimePulse Parameters**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x07	20	*see below*	CK_A CK_B

Payload Contents:

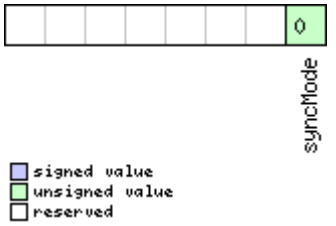
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`interval`	us	Time interval for time pulse
4	U4	-	`length`	us	Length of time pulse
8	I1	-	`status`	-	Time pulse config setting +1 = positive 0 = off -1 = negative
9	U1	-	`timeRef`	-	Alignment to reference time: 0 = UTC time, 1 = GPS time 2 = Local time
10	U1	-	`flags`	-	Bitmask (see graphic below)
11	U1	-	`reserved1`	-	Reserved

CFG-TP continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
12	I2	-	`antennaCableDelay`	ns	Antenna Cable Delay
14	I2	-	`rfGroupDelay`	ns	Receiver RF Group Delay
16	I4	-	`userDelay`	ns	User Time Function Delay (positive delay results in earlier pulse)

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`syncMode`	0=Time pulse always synchronized and only available if time is valid 1=Time pulse allowed to be asynchronized and available even when time is not valid

31.26 CFG-USB (0x06 0x1B)

31.26.1 Poll a USB configuration

Message	**CFG-USB**				
Description	**Poll a USB configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Poll Request				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x1B	0	*see below*	CK_A CK_B
No payload					

31.26.2 Get/Set USB Configuration

Message	**CFG-USB**				
Description	**Get/Set USB Configuration**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get/Set				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x06 0x1B	108	*see below*	CK_A CK_B
Payload Contents:					

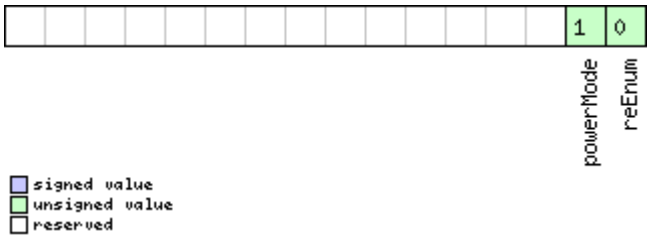
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*

CFG-USB continued

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	U2	-	`reserved1`	-	Always set to zero
6	U2	-	`reserved2`	-	Always set to 1
8	U2	-	`powerConsumption`	-	Power consumed by the device in mA
10	X2	-	`flags`	-	various configuration flags (see graphic below)
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)

32 ESF (0x10)

External Sensor Fusion Messages: i.e. External sensor measurements and status information.

32.1 ESF-MEAS (0x10 0x02)

32.1.1 External Sensor Fusion Measurements (LEA-6R)

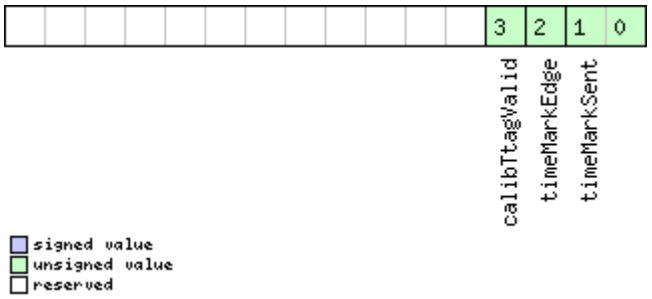
Message	**ESF-MEAS**				
Description	**External Sensor Fusion Measurements (LEA-6R)**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with ADR product variant**).				
Type	Input/Output Message				
Comment	Possible data types for the `data` field are described in section Description of ESF Measurement Data for LEA-6R.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x10 0x02	(8 + 4*N) or (12 + 4*N)	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`timeTag`	-	Time tag of measurement generated by external sensor
4	X2	-	`flags`	-	Flags, set all unused bits to zero (see graphic below)
6	U2	-	`id`	-	identification number of data provider
Start of repeated block (N times)					
8 + 4*N	X4	-	`data`	-	data (see graphic below)
End of repeated block					
Start of optional block					
8 + 4*N	U4	-	`calibTtag`	ms	receiver local time calibrated. This field **must not** be supplied as `calibTtagValid` is set to 0.
End of optional block					

Bitfield flags

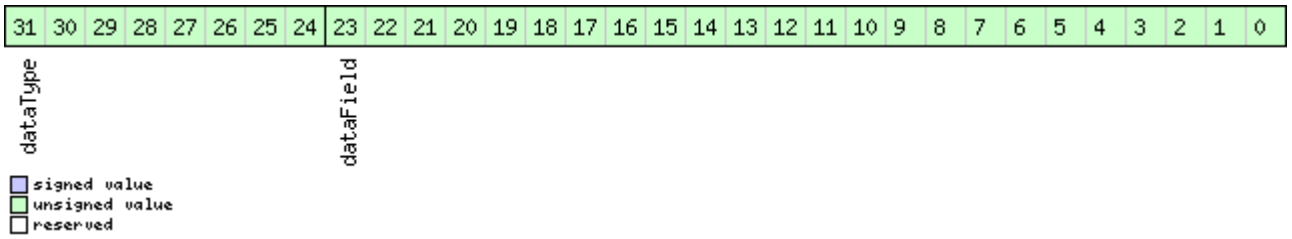
This Graphic explains the bits of `flags`



Name	*Description*
`timeMarkSent`	time mark signal was supplied just prior to sending this message: 0 = none, 1 = on Ext0, 2 = on Ext1
`timeMarkEdge`	trigger on falling (0) or rising (1) edge of time mark signal
`calibTtagValid`	calibration time tag available, always set to zero

Bitfield data

This Graphic explains the bits of `data`



Name	Description
`dataField`	data
`dataType`	type of data (0 = no data; 1..255 = data type)

32.2 ESF-STATUS (0x10 0x10)

32.2.1 Sensor Fusion Status Information (LEA-6R)

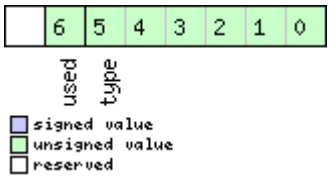
Message	**ESF-STATUS**				
Description	**Sensor Fusion Status Information (LEA-6R)**				
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x10 0x10	16 + 4*numSens	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond Time of week
4	U4	-	`reserved1`	-	Reserved
8	U4	-	`reserved2`	-	Reserved
12	U1	-	`status`	-	Sensor fusion status (0=no fusion; 1=fusion, GNSS and sensor data are used; 2=disabled temporarily, invalid sensor data not used (e.g. car on ferry), 3=disabled permanently (until receiver reset), GNSS-only due to sensor failure)
13	U1	-	`reserved3`	-	Reserved
14	U1	-	`reserved4`	-	Reserved
15	U1	-	`numSens`	-	Number of sensors
Start of repeated block (numSens times)					
16 + 4*N	X1	-	`sensStatus1`	-	the sensor status, part 1 (see graphic below)
17 + 4*N	X1	-	`sensStatus2`	-	the sensor status, part 2 (see graphic below)
18 + 4*N	U1	-	`freq`	Hz	observation frequency
19 + 4*N	U1	-	`reserved6`	-	Reserved
End of repeated block					

Bitfield sensStatus1

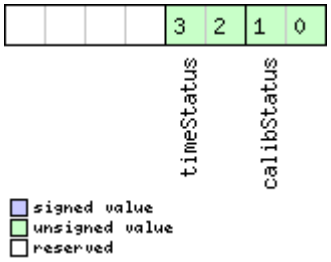
This Graphic explains the bits of `sensStatus1`



Name	*Description*
`type`	sensor type 0: wheel tick front left 1: wheel tick front right 2: wheel tick rear left 3: wheel tick rear right 4: single wheel tick 5: reserved 6: Z-axis gyroscope 7: temperature
`used`	sensor data in current solution flag

Bitfield sensStatus2

This Graphic explains the bits of `sensStatus2`



Name	*Description*
`calibStatus`	00: no calibration 01: calibrating 10: coarse calibration 11: fine calibration
`timeStatus`	00: no data 01: first byte 10: event input 11: tag given

32.2.2 Sensor Fusion Status Information (LEA-6R)

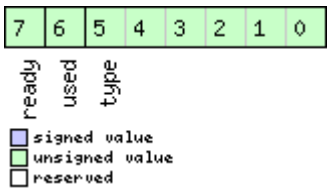
Message	**ESF-STATUS**				
Description	**Sensor Fusion Status Information (LEA-6R)**				
Firmware	Supported on u-blox 6 firmware version 7.03 (**only available with ADR product variant**).				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x10 0x10	16 + 4*numSens	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond Time of week
4	U1	-	`version`	-	Message version (=1 for this version)
5	U1	-	`reserved1`	-	Reserved
6	U2	-	`reserved2`	-	Reserved
8	U4	-	`reserved3`	-	Reserved
12	U1	-	`status`	-	Sensor fusion status (0=no fusion; 1=fusion, GNSS and sensor data are used; 2=disabled temporarily, invalid sensor data not used (e.g. car on ferry), 3=disabled permanently (until receiver reset), GNSS-only due to sensor failure)
13	U1	-	`reserved4`	-	Reserved
14	U1	-	`reserved5`	-	Reserved
15	U1	-	`numSens`	-	Number of sensors
Start of repeated block (numSens times)					
16 + 4*N	X1	-	`sensStatus1`	-	The sensor status, part 1 (see graphic below)
17 + 4*N	X1	-	`sensStatus2`	-	The sensor status, part 2 (see graphic below)
18 + 4*N	U1	-	`freq`	Hz	observation frequency
19 + 4*N	X1	-	`faults`	-	Sensor faults (see graphic below)
End of repeated block					

Bitfield sensStatus1

This Graphic explains the bits of `sensStatus1`



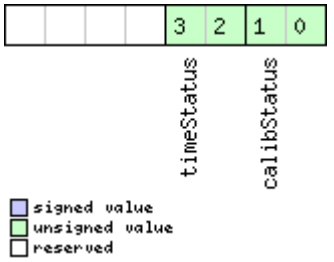
Name	*Description*

Bitfield sensStatus1 Description continued

Name	*Description*
`type`	sensor type 0: wheel tick front left 1: wheel tick front right 2: wheel tick rear left 3: wheel tick rear right 4: single wheel tick 5: reserved 6: Z-axis gyroscope 7: temperature
`used`	The sensor data was used for the current solution
`ready`	The sensor configuration is availabe or not required

Bitfield sensStatus2

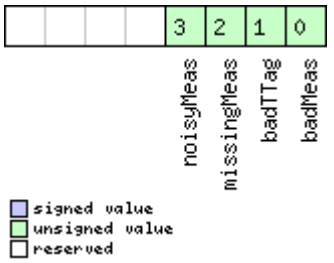
This Graphic explains the bits of `sensStatus2`



Name	*Description*
`calibStatus`	00: No calibration 01: Calibrating, sensor not yet calibrated 10: Calibrating, sensor coarsely calibrated 11: Calibrating, sensor finely calibrated A reasonable DR performance is only possible when at least coarse calibration has been achieved. Depending on the quality of the GNSS signals and the ESF sensor data, fine calibration may take a long time or may even be never obtained.
`timeStatus`	00: No data 01: Reception of the first byte used to tag the measurement 10: Event input used to tag the measurement 11: Time tag provided with the data

Bitfield faults

This Graphic explains the bits of `faults`



Name	*Description*
`badMeas`	Bad measurements seen

Bitfield faults Description continued

Name	*Description*
`badTTag`	Bad measurement ttags seen
`missingMeas`	Measurements missing or misaligned
`noisyMeas`	Measurements noise is high

33 INF (0x04)

Information Messages: i.e. Printf-Style Messages, with IDs such as Error, Warning, Notice.
The INF Class is basically an output class that allows the firmware and application code to output strings with a printf-style call. All INF messages have an associated type to indicate the kind of message.

33.1 INF-DEBUG (0x04 0x04)

33.1.1 ASCII String output, indicating debug output

Message	**INF-DEBUG**				
Description	**ASCII String output, indicating debug output**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	*Header*	*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	-	`char`	-	ASCII Character
End of repeated block					

33.2 INF-ERROR (0x04 0x00)

33.2.1 ASCII String output, indicating an error

Message	**INF-ERROR**				
Description	**ASCII String output, indicating an error**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	*Header*	*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	-	`char`	-	ASCII Character
End of repeated block					

33.3 INF-NOTICE (0x04 0x02)

33.3.1 ASCII String output, with informational contents

Message	**INF-NOTICE**				
Description	**ASCII String output, with informational contents**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	*Header*	*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	-	`char`	-	ASCII Character
End of repeated block					

33.4 INF-TEST (0x04 0x03)

33.4.1 ASCII String output, indicating test output

Message	**INF-TEST**				
Description	**ASCII String output, indicating test output**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	*Header*	*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	-	`char`	-	ASCII Character
End of repeated block					

33.5 INF-WARNING (0x04 0x01)

33.5.1 ASCII String output, indicating a warning

Message	**INF-WARNING**				
Description	**ASCII String output, indicating a warning**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Output				
Comment	This message has a variable length payload, representing an ASCII string.				
Message Structure	*Header*	*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	-	`char`	-	ASCII Character
End of repeated block					

34 MON (0x0A)

Monitoring Messages: i.e. Comunication Status, CPU Load, Stack Usage, Task Status.
Messages in this class are sent to report GPS receiver status, such as CPU load, stack usage, I/O subsystem statistics etc.

34.1 MON-HW2 (0x0A 0x0B)

34.1.1 Extended Hardware Status

Message	**MON-HW2**				
Description	**Extended Hardware Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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` respectively, the better. • Ideally, the magnitude of the I-part (`magI`) and the Q-part (`magQ`) of the complex signal should be the same.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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)
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]	-	`reserved0`	-	Reserved
8	X4	-	`lowLevCfg`	-	Low-level configuration
12	U4[2]	-	`reserved1`	-	Reserved
20	X4	-	`postStatus`	-	POST status word
24	U4	-	`reserved2`	-	Reserved

34.2 MON-HW (0x0A 0x09)

34.2.1 Hardware Status

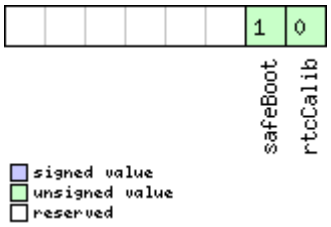
Message	**MON-HW**				
Description	**Hardware Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 6.02.				
Type	Periodic/Polled				
Comment	Status of different aspect of the hardware, such as Antenna, PIO/Peripheral Pins, Noise Level, Automatic Gain Control (AGC)				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0A 0x09	68	*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
24	X4	-	`usedMask`	-	Mask of Pins that are used by the Virtual Pin Manager
28	U1[25]	-	`VP`	-	Array of Pin Mappings for each of the 25 Physical Pins
53	U1	-	`jamInd`	-	Jamming indicator, scaled (0 = no jamming, 255 = strong jamming)
54	U2	-	`reserved3`	-	Reserved
56	X4	-	`pinIrq`	-	Mask of Pins Value using the PIO Irq
60	X4	-	`pullH`	-	Mask of Pins Value using the PIO Pull High Resistor
64	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)

34.2.2 Hardware Status

Message	**MON-HW**				
Description	**Hardware Status**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Periodic/Polled				
Comment	Status of different aspect of the hardware, such as Antenna, PIO/Peripheral Pins, Noise Level, Automatic Gain Control (AGC)				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0A 0x09	68	*see below*	CK_A CK_B

Payload Contents:

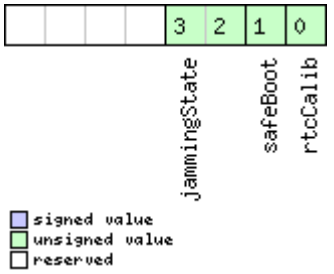
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	X4	-	`pinSel`	-	Mask of Pins Set as Peripheral/PIO
4	X4	-	`pinBank`	-	Mask of Pins Set as Bank A/B
8	X4	-	`pinDir`	-	Mask of Pins Set as Input/Output
12	X4	-	`pinVal`	-	Mask of Pins Value Low/High
16	U2	-	`noisePerMS`	-	Noise Level as 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
24	X4	-	`usedMask`	-	Mask of Pins that are used by the Virtual Pin Manager
28	U1[25]	-	`VP`	-	Array of Pin Mappings for each of the 25 Physical Pins
53	U1	-	`jamInd`	-	CW Jamming indicator, scaled (0 = no CW jamming, 255 = strong CW jamming)
54	U2	-	`reserved3`	-	Reserved
56	X4	-	`pinIrq`	-	Mask of Pins Value using the PIO Irq
60	X4	-	`pullH`	-	Mask of Pins Value using the PIO Pull High Resistor

MON-HW continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
64	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)

34.3 MON-IO (0x0A 0x02)

34.3.1 I/O Subsystem Status

Message	**MON-IO**				
Description	**I/O Subsystem Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	The size of the message is determined by the number of ports 'N' the receiver supports, i.e. on ANTARIS this is always 4, on u-blox 5 the number of ports is 6.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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)					
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	-	`rxBusy`	-	Flag is receiver is busy
17 + 20*N	U1	-	`txBusy`	-	Flag is transmitter is busy
18 + 20*N	U2	-	`reserved1`	-	Reserved
End of repeated block					

34.4 MON-MSGPP (0x0A 0x06)

34.4.1 Message Parse and Process Status

Message	**MON-MSGPP**				
Description	**Message Parse and Process Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*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 target0
16	U2[8]	-	`msg2`	msgs	Number of successfully parsed messages for each protocol on target1
32	U2[8]	-	`msg3`	msgs	Number of successfully parsed messages for each protocol on target2
48	U2[8]	-	`msg4`	msgs	Number of successfully parsed messages for each protocol on target3
64	U2[8]	-	`msg5`	msgs	Number of successfully parsed messages for each protocol on target4
80	U2[8]	-	`msg6`	msgs	Number of successfully parsed messages for each protocol on target5
96	U4[6]	-	`skipped`	bytes	Number skipped bytes for each target

34.5 MON-RXBUF (0x0A 0x07)

34.5.1 Receiver Buffer Status

Message	**MON-RXBUF**				
Description	**Receiver Buffer Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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

34.6 MON-RXR (0x0A 0x21)

34.6.1 Receiver Status Information

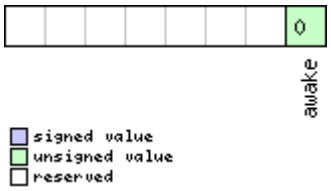
Message	**MON-RXR**				
Description	**Receiver Status Information**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Get				
Comment	The receiver ready message is sent when the receiver changes from or to backup mode.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0A 0x21	1	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`flags`	-	Receiver status flags (see graphic below)

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`awake`	not in Backup mode

34.7 MON-TXBUF (0x0A 0x08)

34.7.1 Transmitter Buffer Status

Message	**MON-TXBUF**				
Description	**Transmitter Buffer Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0A 0x08	28	*see below*	CK_A CK_B

Payload Contents:

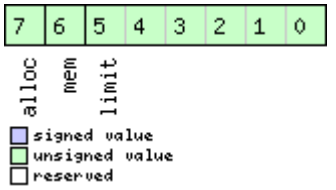
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U2[6]	-	`pending`	bytes	Number of bytes pending in transmitter buffer for each target
12	U1[6]	-	`usage`	%	Maximum usage transmitter buffer during the last sysmon period for each target
18	U1[6]	-	`peakUsage`	%	Maximum usage transmitter buffer for each target
24	U1	-	`tUsage`	%	Maximum usage of transmitter buffer during the last sysmon period for all targets

MON-TXBUF continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
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)

34.8 MON-VER (0x0A 0x04)

34.8.1 Receiver/Software/ROM Version

Message	**MON-VER**				
Description	**Receiver/Software/ROM Version**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Answer to Poll				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0A 0x04	70 + 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
40	CH[30]	-	`romVersion`	-	Zero-terminated ROM Version String
Start of repeated block (N times)					
70 + 30*N	CH[30]	-	`extension`	-	Installed Extension Package Version
End of repeated block					

35 NAV (0x01)

Navigation Results: i.e. Position, Speed, Time, Acc, Heading, DOP, SVs used.
Messages in the NAV Class output Navigation Data such as position, altitude and velocity in a number of formats. Additionally, status flags and accuracy figures are output.

35.1 NAV-AOPSTATUS (0x01 0x60)

35.1.1 AssistNow Autonomous Status

Message	**NAV-AOPSTATUS**				
Description	**AssistNow Autonomous Status**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Periodic/Polled				
Comment	This message provides information on the current availability of *AssistNow Autonomous* data and the current state of the 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x60	20	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS millisecond time of week
4	U1	-	`config`	-	*AssistNow Autonomous* is disabled (0) or enabled (not 0)
5	U1	-	`status`	-	*AssistNow Autonomous* subsystem is idle (0) or running (not 0)
6	U1	-	`reserved0`	-	Always set to zero
7	U1	-	`reserved1`	-	Always set to zero
8	U4	-	`avail`	-	data availability mask for GPS SVs (bits 0-31 correspond to GPS PRN 1-32)
12	U4	-	`reserved2`	-	Always set to zero
16	U4	-	`reserved3`	-	Always set to zero

35.2 NAV-CLOCK (0x01 0x22)

35.2.1 Clock Solution

Message	**NAV-CLOCK**				
Description	**Clock Solution**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*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 Millisecond Time of week

NAV-CLOCK continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
4	I4	-	`clkB`	ns	Clock bias in nanoseconds
8	I4	-	`clkD`	ns/s	Clock drift in nanoseconds per second
12	U4	-	`tAcc`	ns	Time Accuracy Estimate
16	U4	-	`fAcc`	ps/s	Frequency Accuracy Estimate

35.3 NAV-DGPS (0x01 0x31)

35.3.1 DGPS Data Used for NAV

Message	**NAV-DGPS**				
Description	**DGPS Data Used for NAV**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Periodic/Polled				
Comment	This message outputs the Correction data as it has been applied to the current NAV Solution. See also the notes on the RTCM protocol.				
Message Structure	*Header*	*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 Millisecond time of week
4	I4	-	`age`	ms	Age of newest correction data
8	I2	-	`baseId`	-	DGPS Base Station ID
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. - 00: none - 01: PR+PRR Correction
14	U2	-	`reserved1`	-	Reserved
Start of repeated block (numCh times)					
16 + 12*N	U1	-	`svid`	-	Satellite ID
17 + 12*N	U1	-	`flags`	-	Bitmask / Channel Number Bits 0x01 .. 0x08: = GPS Channel this SV is on Bit 0x10: is DGPS Used for this SV/Channel? Bit 0x20 .. 0x80: reserved
18 + 12*N	U2	-	`ageC`	ms	Age of latest correction data
20 + 12*N	R4	-	`prc`	m	Pseudo Range Correction
24 + 12*N	R4	-	`prrc`	m/s	Pseudo Range Rate Correction
End of repeated block					

35.4 NAV-DOP (0x01 0x04)

35.4.1 Dilution of precision

Message	**NAV-DOP**				
Description	**Dilution of precision**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Millisecond Time of Week
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

35.5 NAV-EKFSTATUS (0x01 0x40)

35.5.1 Dead Reckoning Software Status

Message	**NAV-EKFSTATUS**				
Description	**Dead Reckoning Software Status**				
Firmware	Supported on u-blox 6 firmware version 6.00 (**only available with ADR product variant**).				
Type	Periodic/Polled				
Comment	**This message is only provided for backwards compatibility and should not be utilized for future designs. Instead, the messages ESF-STATUS and ESF-MEAS should be used.** For u-blox 6 firmware the gyroscope value (gyroMean) is only output if the gyroscope is used in the navigation solution. This message is only available on LEA-4R and LEA-6R GPS Receivers.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x40	36	*see below*	CK_A CK_B

Payload Contents:

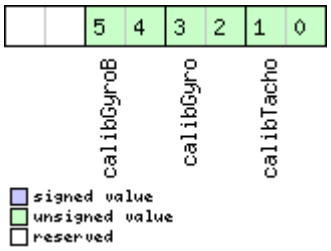
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	I4	-	`pulses`	-	number of pulsed in last update period
4	I4	-	`period`	ms	Duration of last period
8	U4	1e-2	`gyroMean`	-	Uncorrected average Gyro value in last period
12	I2	2^-8	`temperature`	degC	Temperature
14	I1	-	`direction`	-	Direction flag

NAV-EKFSTATUS continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
15	X1	-	`calibStatus`	-	Calibration Status (see graphic below)
16	I4	1e-5	`pulseScale`	-	Current Scale Factor of Speed Pulse
20	I4	1e-5	`gyroBias`	-	Current Bias of Gyro
24	I4	1e-5	`gyroScale`	-	Current Scale Factor of Gyro
28	I2	1e-4	`accPulseScale`	-	Accuracy of Speed Pulse Scale Factor [percentage of initial value]
30	I2	1e-4	`accGyroBias`	-	Accuracy of Bias of Gyro [percentage of initial value]
32	I2	1e-4	`accGyroScale`	-	Accuracy of Scale Factor of Gyro [percentage of initial value]
34	X1	-	`measUsed`	-	Measurements used (see graphic below)
35	U1	-	`reserved2`	-	Reserved

Bitfield calibStatus

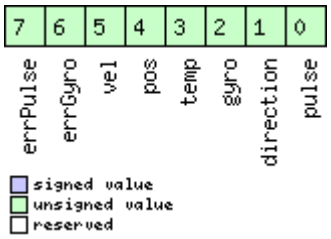
This Graphic explains the bits of `calibStatus`



Name	*Description*
`calibTacho`	Calibration of Scale factor Tacho 00: no calibration 01: calibrating 02: coarse calibration 03: fine calibration
`calibGyro`	Calibration of Scale factor Gyro 00: no calibration 01: calibrating 02: coarse calibration 03: fine calibration
`calibGyroB`	Calibration of Bias Gyro 00: no calibration 01: calibrating 02: coarse calibration 03: fine calibration

Bitfield measUsed

This Graphic explains the bits of `measUsed`



Name	*Description*
`pulse`	Tacho Pulse used
`direction`	forward/backward signal used
`gyro`	Gyro used
`temp`	Temperature used
`pos`	GPS Position used
`vel`	GPS Velocity used
`errGyro`	An inconsistency with the GYRO sensor input was detected. EKF is temporarily disabled. GPS-only data is being output
`errPulse`	An inconsistency with the speed pulse sensor input was detected. EKF is temporarily disabled. GPS-only data is being output

35.6 NAV-POSECEF (0x01 0x01)

35.6.1 Position Solution in ECEF

Message	**NAV-POSECEF**				
Description	**Position Solution in ECEF**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	**See important comments concerning validity of position given in section Navigation Output Filters.** -				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Millisecond Time of Week
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

35.7 NAV-POSLLH (0x01 0x02)

35.7.1 Geodetic Position Solution

Message	**NAV-POSLLH**				
Description	**Geodetic Position Solution**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
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 CFG-DAT.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Millisecond Time of Week
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

35.8 NAV-SBAS (0x01 0x32)

35.8.1 SBAS Status Data

Message	**NAV-SBAS**				
Description	**SBAS Status Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	This message outputs the status of the SBAS sub system				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x32	12 + 12*cnt	*see below*	CK_A CK_B

Payload Contents:

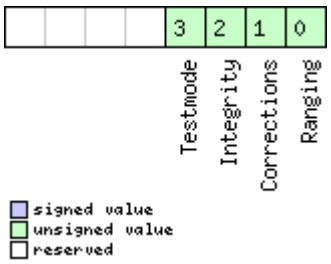
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond time of week
4	U1	-	`geo`	-	PRN Number of the GEO where correction and integrity data is used from
5	U1	-	`mode`	-	SBAS Mode 0 Disabled 1 Enabled Integrity 3 Enabled Testmode

NAV-SBAS continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
6	I1	-	`sys`	-	SBAS System (WAAS/EGNOS/...) -1 Unknown 0 WAAS 1 EGNOS 2 MSAS 16 GPS
7	X1	-	`service`	-	SBAS Services available (see graphic below)
8	U1	-	`cnt`	-	Number of SV data following
9	U1[3]	-	`reserved0`	-	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	-	`reserved1`	-	Reserved
18 + 12*N	I2	-	`prc`	cm	Pseudo Range correction in [cm]
20 + 12*N	U2	-	`reserved2`	-	Reserved
22 + 12*N	I2	-	`ic`	cm	Ionosphere correction in [cm]
End of repeated block					

Bitfield service

This Graphic explains the bits of `service`



35.9 NAV-SOL (0x01 0x06)

35.9.1 Navigation Solution Information

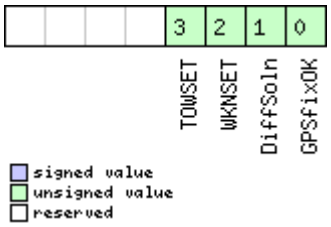
Message	**NAV-SOL**				
Description	**Navigation Solution Information**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	This message combines Position, velocity and time solution in ECEF, including accuracy figures				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Millisecond Time of Week
4	I4	-	`fTOW`	ns	Fractional Nanoseconds remainder of rounded ms above, range -500000 .. 500000
8	I2	-	`week`	-	GPS week (GPS time)
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)
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	U4	-	`reserved2`	-	Reserved

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`GPSfixOK`	i.e within DOP & ACC Masks
`DiffSoln`	1 if DGPS used
`WKNSET`	1 if Week Number valid
`TOWSET`	1 if Time of Week valid

35.10 NAV-STATUS (0x01 0x03)

35.10.1 Receiver Navigation Status

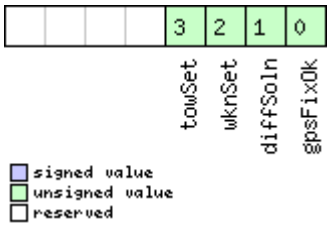
Message	**NAV-STATUS**				
Description	**Receiver Navigation Status**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	**See important comments concerning validity of position and velocity given in section Navigation Output Filters.** -				
Message Structure	*Header*	*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 Millisecond Time of Week
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`	-	Time to first fix (millisecond time tag)
12	U4	-	`msss`	-	Milliseconds since Startup / Reset

Bitfield flags

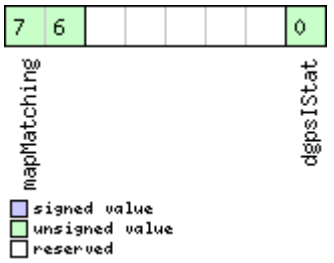
This Graphic explains the bits of `flags`



Name	*Description*
`gpsFixOk`	position and velocity valid and within DOP and ACC Masks, see also important comments in section Navigation Output Filters.
`diffSoln`	1 if DGPS used
`wknSet`	1 if Week Number valid
`towSet`	1 if Time of Week valid

Bitfield fixStat

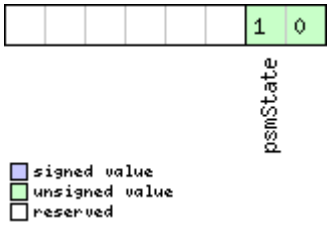
This Graphic explains the bits of `fixStat`



Name	*Description*
`dgpsIStat`	DGPS Input Status 0: none 1: PR+PRR Correction
`mapMatching`	map matching status, see section Map Matching Input for details. 00: none 01: valid, i.e. map matching data was received, but was too old 10: used, map matching data was applied 11: DR, map matching was the reason to enable the dead reckoning `gpsFix` type instead of publishing no fix

Bitfield flags2

This Graphic explains the bits of `flags2`



Name	*Description*
`psmState`	power safe mode state (0=ACQUISITION [or when psm disabled], 1=TRACKING, 2=POWER OPTIMIZED TRACKING, 3=INACTIVE). Only for FW version >= 7.01; undefined otherwise.

35.11 NAV-SVINFO (0x01 0x30)

35.11.1 Space Vehicle Information

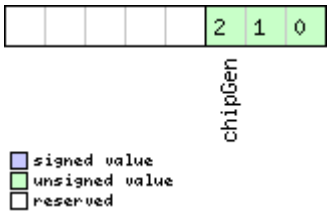
Message	**NAV-SVINFO**				
Description	**Space Vehicle Information**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x30	8 + 12*numCh	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond time of week
4	U1	-	`numCh`	-	Number of channels
5	X1	-	`globalFlags`	-	Bitmask (see graphic below)
6	U2	-	`reserved2`	-	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
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 centimetres
End of repeated block					

Bitfield globalFlags

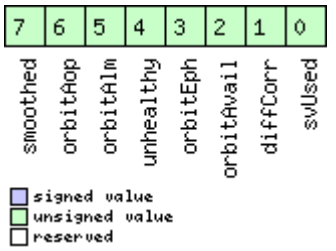
This Graphic explains the bits of `globalFlags`



Name	*Description*
`chipGen`	Chip hardware generation 0: Antaris, Antaris 4 1: u-blox 5 2: u-blox 6

Bitfield flags

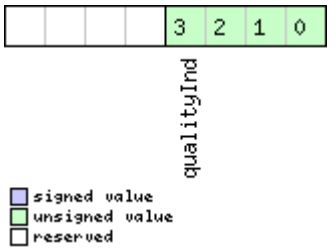
This Graphic explains the bits of `flags`



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 Almanach)
`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 (see PPP for details)

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: This channel is idle 1: Channel is searching 2: Signal aquired 3: Signal detected but unusable 4: Code Lock on Signal 5, 6, 7: Code and Carrier locked

35.12 NAV-TIMEGPS (0x01 0x20)

35.12.1 GPS Time Solution

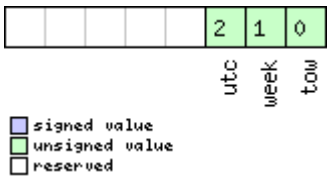
Message	**NAV-TIMEGPS**				
Description	**GPS Time Solution**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x20	16	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond time of Week
4	I4	-	`fTOW`	ns	Fractional Nanoseconds remainder of rounded ms above, range -500000 .. 500000
8	I2	-	`week`	-	GPS week (GPS time)
10	I1	-	`leapS`	s	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*
`tow`	1=Valid Time of Week
`week`	1=Valid Week Number
`utc`	1=Valid Leap Seconds, i.e. Leap Seconds already known

35.13 NAV-TIMEUTC (0x01 0x21)

35.13.1 UTC Time Solution

Message	**NAV-TIMEUTC**				
Description	**UTC Time Solution**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	-				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x01 0x21	20	*see below*	CK_A CK_B

Payload Contents:

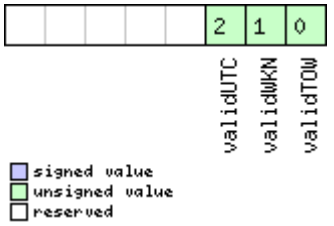
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`iTOW`	ms	GPS Millisecond Time of Week

NAV-TIMEUTC continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
4	U4	-	`tAcc`	ns	Time Accuracy Estimate
8	I4	-	`nano`	ns	Nanoseconds 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..59 (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
`validWKN`	1 = Valid Week Number
`validUTC`	1 = Valid UTC (Leap Seconds already known)

35.14 NAV-VELECEF (0x01 0x11)

35.14.1 Velocity Solution in ECEF

Message	**NAV-VELECEF**				
Description	**Velocity Solution in ECEF**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	**See important comments concerning validity of velocity given in section Navigation Output Filters.** -				
Message Structure	*Header*	*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 Millisecond Time of Week
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

35.15 NAV-VELNED (0x01 0x12)

35.15.1 Velocity Solution in NED

Message	**NAV-VELNED**				
Description	**Velocity Solution in NED**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	**See important comments concerning validity of velocity given in section Navigation Output Filters.** -				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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 Millisecond Time of Week
4	I4	-	`velN`	cm/s	NED north velocity
8	I4	-	`velE`	cm/s	NED east velocity
12	I4	-	`velD`	cm/s	NED down velocity
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

36 RXM (0x02)

Receiver Manager Messages: i.e. Satellite Status, RTC Status.
Messages in Class RXM output status and result data from the Receiver Manager.

36.1 RXM-ALM (0x02 0x30)

36.1.1 Poll GPS Constellation Almanach Data

Message	**RXM-ALM**				
Description	**Poll GPS Constellation Almanach Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Request				
Comment	**This message has an empty payload!** Poll GPS Constellation Data (Almanach) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type RXM-ALM as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 0x30	0	*see below*	CK_A CK_B
No payload					

36.1.2 Poll GPS Constellation Almanach Data for a SV

Message	**RXM-ALM**				
Description	**Poll GPS Constellation Almanach Data for a SV**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Request				
Comment	Poll GPS Constellation Data (Almanach) for an SV by sending this message to the receiver. The receiver will return one message of type RXM-ALM as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 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 Almanach Data (Valid Range: 1 .. 32).

36.1.3 GPS Aiding Almanach Input/Output Message

Message	**RXM-ALM**				
Description	**GPS Aiding Almanach Input/Output Message**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Answer / Periodic				
Comment	**This message is provided considered obsolete, please use AID-ALM instead!** • If the WEEK Value is 0, DWRD0 to DWRD7 are not sent as the almanach is not available for the given SV. • 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 Almanach 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 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 Almanach Data is (Valid Range: 1 .. 32 or 51, 56, 63).
4	U4	-	`week`	-	Issue Date of Almanach (GPS week number)
Start of optional block					
8	U4[8]	-	`dwrd`	-	Almanach Words
End of optional block					

36.2 RXM-EPH (0x02 0x31)

36.2.1 Poll GPS Constellation Ephemeris Data

Message	**RXM-EPH**				
Description	**Poll GPS Constellation Ephemeris Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Request				
Comment	**This message has an empty payload!** Poll GPS Constellation Data (Ephemeris) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type RXM-EPH as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 0x31	0	*see below*	CK_A CK_B
No payload					

36.2.2 Poll GPS Constellation Ephemeris Data for a SV

Message	**RXM-EPH**				
Description	**Poll GPS Constellation Ephemeris Data for a SV**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Request				
Comment	Poll GPS Constellation Data (Ephemeris) for an SV by sending this message to the receiver. The receiver will return one message of type RXM-EPH as defined below.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 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).

36.2.3 GPS Aiding Ephemeris Input/Output Message

Message	**RXM-EPH**				
Description	**GPS Aiding Ephemeris Input/Output Message**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Poll Answer / Periodic				
Comment	**This message is provided considered obsolete, please use AID-EPH 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. • SF1D0 to SF3D7 contain the 24 words following the Hand-Over Word (HOW) from the GPS navigation message, subframes 1 to 3. 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.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 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					

36.3 RXM-PMREQ (0x02 0x41)

36.3.1 Requests a Power Management task

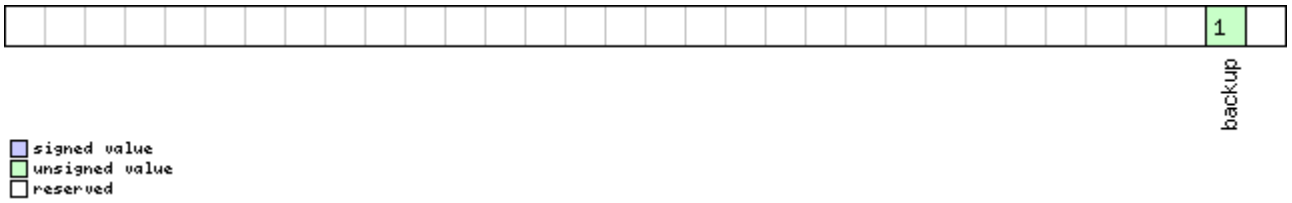
Message	**RXM-PMREQ**				
Description	**Requests a Power Management task**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Input				
Comment	Request of a Power Management related task of the receiver.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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
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

36.4 RXM-RAW (0x02 0x10)

36.4.1 Raw Measurement Data

Message	**RXM-RAW**				
Description	**Raw Measurement Data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Periodic/Polled				
Comment	This message contains all information needed to be able to generate a RINEX file.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 0x10	8 + 24*numSV	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	I4	-	`iTOW`	ms	Measurement integer millisecond GPS time of week (Receiver Time)
4	I2	-	`week`	weeks	Measurement GPS week number (Receiver Time).
6	U1	-	`numSV`	-	# of satellites following.

RXM-RAW continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
7	U1	-	`reserved1`	-	Reserved
Start of repeated block (numSV times)					
8 + 24*N	R8	-	`cpMes`	cycles	Carrier phase measurement [L1 cycles]
16 + 24*N	R8	-	`prMes`	m	Pseudorange measurement [m]
24 + 24*N	R4	-	`doMes`	Hz	Doppler measurement [Hz]
28 + 24*N	U1	-	`sv`	-	Space Vehicle Number
29 + 24*N	I1	-	`mesQI`	-	Nav Measurements Quality Indicator: >=4 : PR+DO OK >=5 : PR+DO+CP OK <6 : likely loss of carrier lock in previous interval
30 + 24*N	I1	-	`cno`	dbHz	Signal strength C/No. (dbHz)
31 + 24*N	U1	-	`lli`	-	Loss of lock indicator (RINEX definition)
End of repeated block					

36.5 RXM-SFRB (0x02 0x11)

36.5.1 Subframe Buffer

Message	**RXM-SFRB**				
Description	**Subframe Buffer**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with raw data product variant**).				
Type	Periodic				
Comment	The content of one single subframe buffer For GPS satellites, the 10 dwrd values contain the parity checked subframe data for 10 Words. Each dwrd has 24 Bits with valid data (Bits 23 to 0). The remaining 8 bits (31 to 24) have an undefined value. The direction within the Word is that the higher order bits are received from the SV first. Example: The Preamble can be found in dwrd[0], at bit position 23 down to 16. For more details on the data format please refer to the ICD-GPS-200C Interface document. For SBAS satellites, the 250 Bit message block can be found in dwrd[0] to dwrd[6] for the first 224 bits. The remaining 26 bits are in dwrd[7], whereas Bits 25 and 24 are the last two data bits, and Bits 23 down to 0 are the parity bits. For more information on SBAS data format, please refer to RTCA/DO-229C (MOPS), Appendix A.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 0x11	42	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`chn`	-	Channel Number
1	U1	-	`svid`	-	ID of Satellite transmitting Subframe
2	X4[10]	-	`dwrd`	-	Words of Data

36.6 RXM-SVSI (0x02 0x20)

36.6.1 SV Status Info

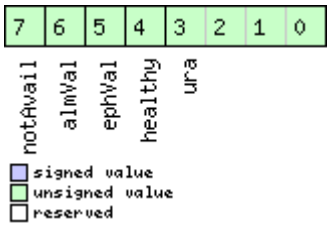
Message	**RXM-SVSI**				
Description	**SV Status Info**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	Status of the receiver manager knowledge about GPS Orbit Validity				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x02 0x20	8 + 6*numSV	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	I4	-	`iTOW`	ms	Measurement integer millisecond GPS time of week
4	I2	-	`week`	weeks	Measurement GPS week number.
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 Almanach and Ephemeris: (see graphic below)
End of repeated block					

Bitfield svFlag

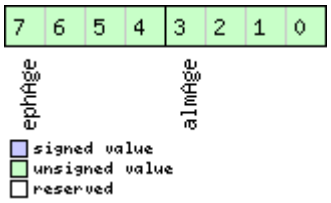
This Graphic explains the bits of `svFlag`



Name	*Description*
`ura`	Figure of Merit (URA) range 0..15
`healthy`	SV healthy flag
`ephVal`	Ephemeris valid
`almVal`	Almanach 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

37 TIM (0x0D)

Timing Messages: i.e. Timepulse Output, Timemark Results.
Messages in this class are output by the receiver, giving information on Timepulse and Timemark measurements.

37.1 TIM-SVIN (0x0D 0x04)

37.1.1 Survey-in data

Message	**TIM-SVIN**				
Description	**Survey-in data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03 (**only available with timing product variant**).				
Type	Periodic/Polled				
Comment	**This message is only supported on timing receivers** This message contains information about survey-in parameters. For details about the Time Mode see section Time Mode Configuration.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0D 0x04	28	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`dur`	s	Passed survey-in observation time
4	I4	-	`meanX`	cm	Current survey-in mean position ECEF X coordinate
8	I4	-	`meanY`	cm	Current survey-in mean position ECEF Y coordinate
12	I4	-	`meanZ`	cm	Current survey-in mean position ECEF Z coordinate
16	U4	-	`meanV`	mm^2	Current survey-in mean position 3D variance
20	U4	-	`obs`	-	Observations used during survey-in
24	U1	-	`valid`	-	Survey-in position validity flag
25	U1	-	`active`	-	Survey-in in progress flag
26	U2	-	`reserved1`	-	Reserved

37.2 TIM-TM2 (0x0D 0x03)

37.2.1 Time mark data

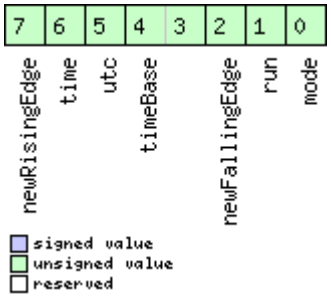
Message	**TIM-TM2**				
Description	**Time mark data**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	This message contains information for high precision time stamping / pulse counting. The delay figures and timebase given in CFG-TP are also applied to the time results output in this message.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0D 0x03	28	*see below*	CK_A CK_B
Payload Contents:					

TIM-TM2 continued

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U1	-	`ch`	time	marker channel 0 or 1
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 1=Time base is GPS 2=Time base is UTC
`utc`	0=UTC not available 1=UTC available
`time`	0=Time is not valid 1=Time is valid (Valid GPS fix)
`newRisingEdge`	new rising edge detected

37.3 TIM-TP (0x0D 0x01)

37.3.1 Timepulse Timedata

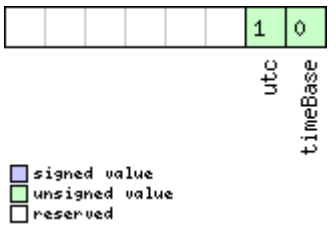
Message	**TIM-TP**				
Description	**Timepulse Timedata**				
Firmware	Supported on u-blox 6 from firmware version 6.00 up to version 7.03.				
Type	Periodic/Polled				
Comment	This message contains information for high precision timing. Note that contents are correct only if the timepulse is set to one pulse per second.				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	0xB5 0x62	0x0D 0x01	16	*see below*	CK_A CK_B

Payload Contents:

Byte Offset	*Number Format*	*Scaling*	*Name*	*Unit*	*Description*
0	U4	-	`towMS`	ms	Timepulse time of week according to time base
4	U4	2^-32	`towSubMS`	ms	Submillisecond part of TOWMS
8	I4	-	`qErr`	ps	Quantization error of timepulse.
12	U2	-	`week`	weeks	Timepulse week number according to time base
14	X1	-	`flags`	-	bitmask (see graphic below)
15	U1	-	`reserved1`	-	Reserved

Bitfield flags

This Graphic explains the bits of `flags`



Name	*Description*
`timeBase`	0=Time base is GPS 1=Time base is UTC
`utc`	0=UTC not available 1=UTC available

37.4 TIM-VRFY (0x0D 0x06)

37.4.1 Sourced Time Verification

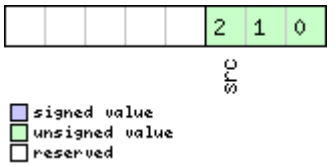
Message	**TIM-VRFY**				
Description	**Sourced Time Verification**				
Firmware	Supported on u-blox 6 firmware version 7.03.				
Type	Polled/Once				
Comment	This message contains verification information about previous time received via AID-INI or from RTC				
Message Structure	*Header*	*ID*	*Length (Bytes)*	*Payload*	*Checksum*
	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

RTCM Protocol

38 Introduction

The RTCM (Radio Technical Commission for Maritime Services) protocol is a unidirectional protocol (input to the receiver) that is used to supply the GPS receiver with real-time differential correction data (DGPS). The RTCM protocol specification is available from http://www.rtcm.org.

39 Supported Messages

Starting with firmware version 7.01, u-blox 6 GPS Technology supports the following RTCM 2.3 messages:

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

40 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.

41 Output

DGPS mode will result in following modified output:

- NMEA-GGA: The NMEA fix status will be 2 ("DPGS"), The age of DGPS corrections and Reference station id will be set.
- NMEA-GLL, NMEA-RMC: The NMEA mode indicator will be D ("Differential").
- NMEA-PUBX00: The status will be D2/D3; The age of DGPS corrections will be set.
- UBX-NAV-SOL: The DGPS will be set.
- UBX-NAV-STATUS: The DGPS will be set; The DGPS input will be set to "PR+PRR".
- 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: DGPS baseline big: 330.3km"

42 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*.

43 Reference

The u-blox 6 RTCM support was implemented according to RTCM 10402.3 ("RECOMMENDED STANDARDS FOR DIFFERENTIAL GNSS").

Appendix

A u-blox 6 Default Settings

The default settings listed in this section apply from u-blox 6 ROM-based receivers with ROM version 6.02 and above. These values assume that the default levels of the configuration pins have been left unchanged. Default settings are dependent on the configuration pin settings, for information regarding these settings, consult the applicable Data Sheet.

A.1 Antenna Supervisor Settings (UBX-CFG-ANT)

For parameter and protocol description see section UBX-CFG-ANT.

Antenna Settings

Parameter	*Default Setting*	*Unit*
Enable Control Signal	Enabled	
Enable Short Circuit Detection	Enabled	
Enable Short Circuit Power Down logic	Enabled	
Enable Automatic Short Circuit Recovery logic	Enabled	
Enable Open Circuit Detection	Disabled	

A.2 Datum Settings (UBX-CFG-DAT)

For parameter and protocol description see section UBX-CFG-DAT.

Datum Default Settings

Parameter	*Default Setting*	*Unit*
Datum	0 – WGS84	

A.3 Navigation Settings (UBX-CFG-NAV5)

For parameter and protocol description see section UBX-CFG-NAV5.

Navigation Default Settings

Parameter	*Default Setting*	*Unit*
Dynamic Platform Model	0 – Portable	
Fix Mode	Auto 2D/3D	#
Fixed Altitude	N/A	m
Fixed Altitude Variance	N/A	m^2
Min SV Elevation	5	deg
DR Timeout	0	s
PDOP Mask	25	-
TDOP Mask	25	-
P Accuracy	100	m
T Accuracy	300	m
Static Hold Threshold	0.00	m/s

The Dynamic Platform Model default setting is different in a firmware with certain premium features enabled. See table below for details.

Dynamic Platform Model Default Setting Variations

Firmware Variant	*Default Setting*
Standard	0 - Portable
Timing Feature Enabled (LEA-6T)	1 - Stationary
Automotive Dead Reckoning Enabled (ADR)	3 - Automotive

A.4 Navigation Settings (UBX-CFG-NAVX5)

For parameter and protocol description see section UBX-CFG-NAVX5.

Navigation Default Settings

Parameter	*Default Setting*	*Unit*
Apply min/max SVs settings	Enabled	
Apply minimum C/N0 settings	Enabled	
Apply initial 3D fix settings	Enabled	
Apply GPS weeknumber rollover settings	Enabled	
Minimum number of SV	3	
Maximum number of SV	16	
Minimum C/N0 for navigation (up to firmware 6.02)	10	dBHz
Minimum C/N0 for navigation (as of firmware 7.01)	7	dBHz
Initial Fix must be 3D	Disabled	
Use *AssistNow Autonomous*	Disabled	
Weeknumber rollover	1603 (u-blox 6 FW7)	

The minimun number of SV default setting is set to 1 in a firmware with the timing premium feature enabled (LEA-6T).

A.5 Output Rates (UBX-CFG-RATE)

For parameter and protocol description see section UBX-CFG-RATE.

Output Rate Default Settings

Parameter	*Default Setting*	*Unit*
Time Source	1 – GPS time	
Measurement Period	1000	ms
Measurement Rate	1	Cycles

A.6 Fix Now Configuration (UBX-CFG-FXN)

Starting with u-blox 6 FW 6.00.

For parameter and protocol description see section UBX-CFG-FXN.

Fix Now Configuration Default Settings

Parameter	*Default Setting*	*Unit*
Sleep	Disabled	
Absolute Alignment	Enabled	
Use on/off time	Disabled	
Re-acquire time	0	ms
Acquire time	0	ms
Off time if re-acquisition failed	10000	ms
Off time if acquisition failed	10000	ms
On time	2000	ms
Off time	n/a	ms
Base TOW	0	ms

A.7 Power Management Configuration (UBX-CFG-PM)

For parameter and protocol description see section UBX-CFG-PM.

Power Management Configuration Default Settings

Parameter	*Default Setting*	*Unit*
Version	0	
EXTINT pin selection	EXTINT0	
EXTINT pin control - keep awake	Disabled	
EXTINT pin control - force backup	Disabled	
Limit peak current	Disabled	
Wait for time fix	Disabled	
Update Real Time Clock	Disabled	
Update ephemeris	Enabled	
Update period	1000	ms
Search period	10000	ms
Grid offset	0	ms
On time	2	s
Minimum acquisition time	0	s

A.8 Power Management 2 Configuration (UBX-CFG-PM2)

For parameter and protocol description see section UBX-CFG-PM2.

Power Management 2 Configuration Default Settings

Parameter	*Default Setting*	*Unit*
Version	1	
EXTINT pin selection	EXTINT0	
EXTINT pin control - keep awake	Disabled	
EXTINT pin control - force backup	Disabled	
Limit peak current	Disabled	
Wait for time fix	Disabled	
Update Real Time Clock	Disabled	

Power Management 2 Configuration Default Settings continued

Parameter	*Default Setting*	*Unit*
Update ephemeris	Enabled	
Do not enter 'inactive for search' state when no fix	Disabled	
Mode of operation	Cyclic tracking	
Update period	1000	ms
Search period	10000	ms
Grid offset	0	ms
On time	2	s
Minimum acquisition time	0	s

A.9 Receiver Manager Configuration (UBX-CFG-RXM)

For parameter and protocol description see section UBX-CFG-RXM.

Power Management Default Settings

Parameter	*Default Setting*	*Unit*
Low power mode	0 - max performance mode	

A.10 SBAS Configuration (UBX-CFG-SBAS)

For parameter and protocol description see section UBX-CFG-SBAS.

SBAS Configuration Default Settings

Parameter	*Default Setting*	*Unit*
SBAS Subsystem	Enabled	
Allow test mode usage	Disabled	
Ranging (Use SBAS for navigation)	Enabled	
Apply SBAS Correction Data	Enabled	
Apply integrity information	Disabled	
Number of search channels	3	
PRN Codes (up to firmware 6.02)	120, 122, 124, 126-127, 129, 131, 134-135, 137-138	
PRN Codes (as of firmware 7.01)	120, 124, 126, 129, 133-134, 137-138	

A.11 Port Setting (UBX-CFG-PRT)

For parameter and protocol description see section UBX-CFG-PRT.

Port Default Settings

Parameter	*Default Setting*	*Unit*
All ports		
Extended TX timeout	0 - disabled	
TX-ready feature	0 - disabled	
DDC/I2C (Target0)		
Protocol in	0+1+2 – UBX+NMEA+RTCM	
Protocol out	0+1 – UBX+NMEA	
USART1 (Target1)		
Protocol in	0+1+2 – UBX+NMEA+RTCM	
Protocol out	0+1 – UBX+NMEA	
Baudrate	9600	baud

Port Default Settings continued

Parameter	*Default Setting*	*Unit*
USART2 (Target2)		
Protocol in	None	
Protocol out	None	
Baudrate	9600	baud
USB (Target3)		
Protocol in	0+1+2 – UBX+NMEA+RTCM	
Protocol out	0+1 – UBX+NMEA	
SPI (Target4)		
Protocol in	0+1+2 – UBX+NMEA+RTCM	
Protocol out	0+1 – UBX+NMEA	

A.12 Port Setting (UBX-CFG-USB)

For parameter and protocol description see section UBX-CFG-USB.

USB default settings

Parameter	*Default Setting*	*Unit*
Power Mode		
Power Mode	Bus powered	
Bus Current required	100	mA

A.13 Message Settings (UBX-CFG-MSG)

For parameter and protocol description see section UBX-CFG-MSG.

Enabled output messages

Message	*Type*	*All Targets*
NMEA - GGA	Out	1
NMEA - GLL	Out	1
NMEA - GSA	Out	1
NMEA - GSV	Out	1
NMEA - RMC	Out	1
NMEA - VTG	Out	1

A.14 NMEA Protocol Settings (UBX-CFG-NMEA)

For parameter and protocol description see section UBX-CFG-NMEA.

NMEA Protocol Default Settings

Parameter	*Default Setting*	*Unit*
Enable position output even for invalid fixes	Disabled	
Enable position even for masked fixes	Disabled	
Enable time output even for invalid times	Disabled	
Enable time output even for invalid dates	Disabled	
Version	2.3	
Compatibility Mode	Disabled	
Consideration Mode	Enabled	
Number of SV	Unlimited	

A.15 INF Messages Settings (UBX-CFG-INF)

For parameter and protocol description see section UBX-CFG-INF.

NMEA default enabled INF msg

Message	*Type*	*All Targets*	*Range/Remark*
INF-Error	Out	1	In NMEA Protocol only (GPTXT)
INF-Warning	Out	1	In NMEA Protocol only (GPTXT)
INF-Notice	Out	1	In NMEA Protocol only (GPTXT)
INF-Test	Out		
INF-Debug	Out		
INF-User	Out	1	In NMEA Protocol only (GPTXT)

A.16 Timepulse Settings (UBX-CFG-TP)

For parameter and protocol description see section UBX-CFG-TP.

TIMEPULSE default settings

Parameter	*Default Setting*	*Unit*
Pulse Mode	+1 – rising	
Pulse Period	1000	ms
Pulse Length	100	ms
Time Source	1 – GPS time	
Cable Delay	50	ns
User Delay	0	ns
SyncMode	0 (no time pulse in case of no fix)	

A.17 Timepulse Settings (UBX-CFG-TP5)

This message applies to u-blox 6.

For parameter and protocol description see section UBX-CFG-TP5.

TIMEPULSE default settings

Parameter	*Default Setting*	*Unit*
Cable Delay	50	ns
RF Groupdelay	0	ns
Period	1000000	us
Period Locked	1000000	us
Pulse Length	0	us
Pulse Length Locked	100000	us
User Delay	0	ns
Timegrid	1 (GPS Time)	
Polarity	1 (rising edge at top of second)	
Align to TOW	1	
IsLength	1	
IsFreq	0	
Locked other setting	1	
Lock to GPS freq	1	
Active	1	

TIMEPULSE2 default settings

Parameter	*Default Setting*	*Unit*
Cable Delay	50	ns
RF Groupdelay	0	ns
Frequency	4	Hz
Frequency Locked	1	Hz
Pulse Length	125000	us
Pulse Length Locked	100000	us
User Delay	0	ns
Timegrid	1 (GPS Time)	
Polarity	1 (rising edge at top of second)	
Align to TOW	1	
IsLength	1	
IsFreq	1	
Locked other setting	1	
Lock to GPS freq	1	
Active	0	

A.18 Jammer/Interference Monitor (UBX-CFG-ITFM)

This message applies to u-blox 6, FW 7.01 and newer.

For parameter and protocol description see section UBX-CFG-ITFM.

Jamming/Interference monitor default settings

Parameter	*Default Setting*	*Unit*
Enable	Disabled	
Broadband interference detection threshold	3	dB
CW interference detection threshold	15	dB

A.19 Remote inventory (UBX-CFG-RINV)

This message applies to u-blox 6, FW 6.00 and newer.

For parameter and protocol description see section UBX-CFG-RINV.

Remote inventory default settings

Parameter	*Default Setting*	*Unit*
Dump data at startup	Disabled	
Data is binary	Disabled	
Data	Notice: no data saved!	

B u-blox 6 Standard firmware versions

Standard FW version strings

Generation	*Version*	*String*
u-blox 6	FW 7.03	ROM CORE 7.03 (45969) Mar 17 2011 16:18:34
u-blox 6		EXT CORE 7.03 (45970) Mar 17 2011 16:26:24 ROM BASE x.xx ...
u-blox 6	FW 7.01	ROM CORE 7.01 (44178) Nov 30 2010 11:40:16
u-blox 6		EXT CORE 7.01 (44179) Nov 30 2010 11:49:29 ROM BASE x.xx ...
u-blox 6	FW 6.02	ROM CORE 6.02 (36023) Oct 15 2009 16:52:08
u-blox 6		EXT CORE 6.02 (36023) Oct 15 2009 16:51:54 ROM BASE x.xx ...

C Geodetic Datum

C.1 Predefined Datum

The following, predefined datum values are available up to firmware version 6.02 and can be configured using the CFG-DAT message. The use of these standard datums is deprecated and is not supported anymore starting with firmware version 7.01. Instead, the other variant of the CFG-DAT message must be used, where the parameters are set directly by the user.

For the ellipsoid parameters, see ellipsoid section below. For the rotation and scale parameters, see rotation and scale section below.

The receiver defaults to WGS84 datum

Geodetic Datum Defined in Firmware

Index	*Description*	*Short*	*Ellipsoid Index*	*Rotation, Scale*	*dX [m]*	*dY [m]*	*dZ [m]*
0	World Geodetic System - 84	WGS84	0	0	0.0	0.0	0.0
1	World Geodetic System - 72	WGS72	23	1	0.0	0.0	4.5
2	Earth-90 - GLONASS Coordinate system	ETH90	8	0	0.0	0.0	4.0
3	Adindan - Mean Solution (Ethiopia & Sudan)	ADI-M	7	0	-166.0	-15.0	204.0
4	Adindan - Burkina Faso	ADI-E	7	0	-118.0	-14.0	218.0
5	Adindan - Cameroon	ADI-F	7	0	-134.0	-2.0	210.0
6	Adindan - Ethiopia	ADI-A	7	0	-165.0	-11.0	206.0
7	Adindan - Mali	ADI-C	7	0	-123.0	-20.0	220.0
8	Adindan - Senegal	ADI-D	7	0	-128.0	-18.0	224.0
9	Adindan - Sudan	ADI-B	7	0	-161.0	-14.0	205.0
10	Afgooye - Somalia	AFG	21	0	-43.0	-163.0	45.0
11	ARC 1950 - Mean (Botswana, Lesotho, Malawi, Swaziland, Zaire, Zambia, Zimbabwe)	ARF-M	7	0	-143.0	-90.0	-294.0
12	ARC 1950 - Botswana	ARF-A	7	0	-138.0	-105.0	-289.0
13	ARC 1950 - Burundi	ARF-H	7	0	-153.0	-5.0	-292.0
14	ARC 1950 - Lesotho	ARF-B	7	0	-125.0	-108.0	-295.0
15	ARC 1950 - Malawi	ARF-C	7	0	-161.0	-73.0	-317.0
16	ARC 1950 - Swaziland	ARF-D	7	0	-134.0	-105.0	-295.0
17	ARC 1950 - Zaire	ARF-E	7	0	-169.0	-19.0	-278.0
18	ARC 1950 - Zambia	ARF-F	7	0	-147.0	-74.0	-283.0
19	ARC 1950 - Zimbabwe	ARF-G	7	0	-142.0	-96.0	-293.0
20	ARC 1960 - Mean (Kenya, Tanzania)	ARS	7	0	-160.0	-6.0	-302.0
21	Ayabelle Lighthouse - Djibouti	PHA	7	0	-79.0	-129.0	145.0
22	Bissau - Guinea-Bissau	BID	20	0	-173.0	253.0	27.0
23	Cape - South Africa	CAP	7	0	-136.0	-108.0	-292.0
24	Carthage - Tunisia	CGE	7	0	-263.0	6.0	431.0
25	Dabola - Guinea	DAL	7	0	-83.0	37.0	124.0
26	Leigon - Ghana	LEH	7	0	-130.0	29.0	364.0
27	Liberia 1964	LIB	7	0	-90.0	40.0	88.0
28	Massawa - Eritrea (Ethiopia)	MAS	5	0	639.0	405.0	60.0
29	Merchich - Morocco	MER	7	0	31.0	146.0	47.0
30	Minna - Cameroon	MIN-A	7	0	-81.0	-84.0	115.0
31	Minna - Nigeria	MIN-B	7	0	-92.0	-93.0	122.0

Geodetic Datum Defined in Firmware continued

Index	Description	Short	Ellipsoid Index	Rotation, Scale	dX [m]	dY [m]	dZ [m]
32	M'Poraloko - Gabon	MPO	7	0	-74.0	-130.0	42.0
33	North Sahara 1959 - Algeria	NSD	7	0	-186.0	-93.0	310.0
34	Old Egyptian 1907 - Egypt	OEG	17	0	-130.0	110.0	-13.0
35	Point 58 - Mean Solution (Burkina Faso & Niger)	PTB	7	0	-106.0	-129.0	165.0
36	Pointe Noire 1948 - Congo	PTN	7	0	-148.0	51.0	-291.0
37	Schwarzeck - Namibia	SCK	5	0	616.0	97.0	-251.0
38	Voirol 1960 - Algeria	VOR	7	0	-123.0	-206.0	219.0
39	Ain El Abd 1970 - Bahrain Island	AIN-A	20	0	-150.0	-250.0	-1.0
40	Ain El Abd 1970 - Saudi Arabia	AIN-B	20	0	-143.0	-236.0	7.0
41	Djakarta (Batavia)- Sumatra (Indonesia)	BAT	5	0	-377.0	681.0	-50.0
42	Hong Kong 1963 - Hong Kong	HKD	20	0	-156.0	-271.0	-189.0
43	Hu-Tzu-Shan - Taiwan	HTN	20	0	-637.0	-549.0	-203.0
44	Indian - Bangladesh	IND-B	9	0	282.0	726.0	254.0
45	Indian - India & Nepal	IND-I	11	0	295.0	736.0	257.0
46	Indian 1954 - Thailand	INF-A	9	0	217.0	823.0	299.0
47	Indian 1960 - Vietnam (near 16N)	ING-A	9	0	198.0	881.0	317.0
48	Indian 1960 - Con Son Island (Vietnam)	ING-B	9	0	182.0	915.0	344.0
49	Indian 1975 - Thailand	INH-A	9	0	209.0	818.0	290.0
50	Indonesian 1974	IDN	19	0	-24.0	-15.0	5.0
51	Kandawala - Sri Lanka	KAN	9	0	-97.0	787.0	86.0
52	Kertau 1948 - West Malaysia & Singapore	KEA	13	0	-11.0	851.0	5.0
53	Nahrwan - Masirah Island (Oman)	NAH-A	7	0	-247.0	-148.0	369.0
54	Nahrwan - United Arab Emirates	NAH-B	7	0	-249.0	-156.0	381.0
55	Nahrwan - Saudi Arabia	NAH-C	7	0	-243.0	-192.0	477.0
56	Oman	FAH	7	0	-346.0	-1.0	224.0
57	Qatar National - Qatar	QAT	20	0	-128.0	-283.0	22.0
58	South Asia - Singapore	SOA	15	0	7.0	-10.0	-26.0
59	Timbalai 1948 - Brunei & East Malaysia (Sarawak & Sabah)	TIL	10	0	-679.0	669.0	-48.0
60	Tokyo - Mean Solution (Japan,Okinawa & South Korea)	TOY-M	5	0	-148.0	507.0	685.0
61	Tokyo - Japan	TOY-A	5	0	-148.0	507.0	685.0
62	Tokyo - Okinawa	TOY-C	5	0	-158.0	507.0	676.0
63	Tokyo - South Korea	TOY-B	5	0	-146.0	507.0	687.0
64	Australian Geodetic 1966 - Australia & Tasmania	AUA	3	0	-133.0	-48.0	148.0
65	Australian Geodetic 1984 - Australia & Tasmania	AUG	3	0	-134.0	-48.0	149.0
66	European 1950 - Mean (AU, B, DK, FN, F, G, GR, I, LUX, NL, N, P, E, S, CH)	EUR-M	20	0	-87.0	-98.0	-121.0
67	European 1950 - Western Europe (AU, DK, FR, G, NL, CH)	EUR-A	20	0	-87.0	-96.0	-120.0
68	European 1950 - Cyprus	EUR-E	20	0	-104.0	-101.0	-140.0
69	European 1950 - Egypt	EUR-F	20	0	-130.0	-117.0	-151.0

Geodetic Datum Defined in Firmware continued

Index	*Description*	*Short*	*Ellipsoid Index*	*Rotation, Scale*	*dX [m]*	*dY [m]*	*dZ [m]*
70	European 1950 - England, Wales, Scotland & Channel Islands	EUR-G	20	0	-86.0	- 96.0	-120.0
71	European 1950 - England, Wales, Scotland & Ireland	EUR-K	20	0	-86.0	- 96.0	-120.0
72	European 1950 - Greece	EUR-B	20	0	-84.0	-95.0	-130.0
73	European 1950 - Iran	EUR-H	20	0	-117.0	-132.0	-164.0
74	European 1950 - Italy - Sardinia	EUR-I	20	0	-97.0	-103.0	-120.0
75	European 1950 - Italy - Sicily	EUR-J	20	0	-97.0	-88.0	-135.0
76	European 1950 - Malta	EUR-L	20	0	-107.0	-88.0	-149.0
77	European 1950 - Norway & Finland	EUR-C	20	0	-87.0	-95.0	-120.0
78	European 1950 - Portugal & Spain	EUR-D	20	0	-84.0	-107.0	-120.0
79	European 1950 - Tunisia	EUR-T	20	0	-112.0	-77.0	-145.0
80	European 1979 - Mean Solution (AU, FN, NL, N, E, S, CH)	EUS	20	0	-86.0	-98.0	-119.0
81	Hjorsey 1955 - Iceland	HJO	20	0	-73.0	46.0	-86.0
82	Ireland 1965	IRL	2	0	506.0	-122.0	611.0
83	Ordnance Survey of GB 1936 - Mean (E, IoM, S, ShI, W)	OGB-M	1	0	375.0	-111.0	431.0
84	Ordnance Survey of GB 1936 - England	OGB-A	1	0	371.0	-112.0	434.0
85	Ordnance Survey of GB 1936 - England, Isle of Man & Wales	OGB-B	1	0	371.0	-111.0	434.0
86	Ordnance Survey of GB 1936 - Scotland & Shetland Isles	OGB-C	1	0	384.0	-111.0	425.0
87	Ordnance Survey of GB 1936 - Wales	OGB-D	1	0	370.0	-108.0	434.0
88	Rome 1940 - Sardinia Island	MOD	20	0	-225.0	-65.0	9.0
89	S-42 (Pulkovo 1942) - Hungary	SPK	21	0	28.0	-121.0	-77.0
90	S-JTSK Czechoslavakia (prior to 1 Jan 1993)	CCD	5	0	589.0	76.0	480.0
91	Cape Canaveral - Mean Solution (Florida & Bahamas)	CAC	6	0	-2.0	151.0	181.0
92	N. American 1927 - Mean Solution (CONUS)	NAS-C	6	0	-8.0	160.0	176.0
93	N. American 1927 - Western US	NAS-B	6	0	-8.0	159.0	175.0
94	N. American 1927 - Eastern US	NAS-A	6	0	-9.0	161.0	179.0
95	N. American 1927 - Alaska (excluding Aleutian Islands)	NAS-D	6	0	-5.0	135.0	172.0
96	N. American 1927 - Aleutian Islands, East of 180W	NAS-V	6	0	-2.0	152.0	149.0
97	N. American 1927 - Aleutian Islands, West of 180W	NAS-W	6	0	2.0	204.0	105.0
98	N. American 1927 - Bahamas (excluding San Salvador Island)	NAS-Q	6	0	-4.0	154.0	178.0
99	N. American 1927 - San Salvador Island	NAS-R	6	0	1.0	140.0	165.0
100	N. American 1927 - Canada Mean Solution (including Newfoundland)	NAS-E	6	0	-10.0	158.0	187.0
101	N. American 1927 - Alberta & British Columbia	NAS-F	6	0	-7.0	162.0	188.0

Geodetic Datum Defined in Firmware continued

Index	*Description*	*Short*	*Ellipsoid Index*	*Rotation, Scale*	*dX [m]*	*dY [m]*	*dZ [m]*
102	N. American 1927 - Eastern Canada (Newfoundland, New Brunswick, Nova Scotia & Quebec)	NAS-G	6	0	-22.0	160.0	190.0
103	N. American 1927 - Manitoba & Ontario	NAS-H	6	0	-9.0	157.0	184.0
104	N. American 1927 - Northwest Territories & Saskatchewan	NAS-I	6	0	4.0	159.0	188.0
105	N. American 1927 - Yukon	NAS-J	6	0	-7.0	139.0	181.0
106	N. American 1927 - Canal Zone	NAS-O	6	0	0.0	125.0	201.0
107	N. American 1927 - Caribbean	NAS-P	6	0	-3.0	142.0	183.0
108	N. American 1927 - Central America	NAS-N	6	0	0.0	125.0	194.0
109	N. American 1927 - Cuba	NAS-T	6	0	-9.0	152.0	178.0
110	N. American 1927 - Greenland (Hayes Peninsula)	NAS-U	6	0	11.0	114.0	195.0
111	N. American 1927 - Mexico	NAS-L	6	0	-12.0	130.0	190.0
112	N. American 1983 - Alaska (excluding Aleutian Islands)	NAR-A	16	0	0.0	0.0	0.0
113	N. American 1983 - Aleutian Islands	NAR-E	16	0	-2.0	0.0	4.0
114	N. American 1983 - Canada	NAR-B	16	0	0.0	0.0	0.0
115	N. American 1983 - Mean Solution (CONUS)	NAR-C	16	0	0.0	0.0	0.0
116	N. American 1983 - Hawaii	NAR-H	16	0	1.0	1.0	-1.0
117	N. American 1983 - Mexico & Central America	NAR-D	16	0	0.0	0.0	0.0
118	Bogota Observatory - Colombia	BOO	20	0	307.0	304.0	-318.0
119	Campo Inchauspe 1969 - Argentina	CAI	20	0	-148.0	136.0	90.0
120	Chua Astro - Paraguay	CHU	20	0	-134.0	229.0	-29.0
121	Corrego Alegre - Brazil	COA	20	0	-206.0	172.0	-6.0
122	Prov S. American 1956 - Mean Solution (Bol, Col, Ecu, Guy, Per & Ven)	PRP-M	20	0	-288.0	175.0	-376.0
123	Prov S. American 1956 - Bolivia	PRP-A	20	0	-270.0	188.0	-388.0
124	Prov S. American 1956 - Northern Chile (near 19S)	PRP-B	20	0	-270.0	183.0	-390.0
125	Prov S. American 1956 - Southern Chile (near 43S)	PRP-C	20	0	-305.0	243.0	-442.0
126	Prov S. American 1956 - Colombia	PRP-D	20	0	-282.0	169.0	-371.0
127	Prov S. American 1956 - Ecuador	PRP-E	20	0	-278.0	171.0	-367.0
128	Prov S. American 1956 - Guyana	PRP-F	20	0	-298.0	159.0	-369.0
129	Prov S. American 1956 - Peru	PRP-G	20	0	-279.0	175.0	-379.0
130	Prov S. American 1956 - Venezuela	PRP-H	20	0	-295.0	173.0	-371.0
131	Prov South Chilean 1963	HIT	20	0	16.0	196.0	93.0
132	South American 1969 - Mean Solution (Arg, Bol, Bra, Chi, Col, Ecu, Guy, Par, Per, Tri & Tob, Ven)	SAN-M	22	0	-57.0	1.0	-41.0
133	South American 1969 - Argentina	SAN-A	22	0	-62.0	-1.0	-37.0
134	South American 1969 - Bolivia	SAN-B	22	0	-61.0	2.0	-48.0
135	South American 1969 - Brazil	SAN-C	22	0	-60.0	-2.0	-41.0
136	South American 1969 - Chile	SAN-D	22	0	-75.0	-1.0	-44.0

Geodetic Datum Defined in Firmware continued

Index	*Description*	*Short*	*Ellipsoid Index*	*Rotation, Scale*	*dX [m]*	*dY [m]*	*dZ [m]*
137	South American 1969 - Colombia	SAN-E	22	0	-44.0	6.0	-36.0
138	South American 1969 - Ecuador (excluding Galapagos Islands)	SAN-F	22	0	-48.0	3.0	-44.0
139	South American 1969 - Baltra, Galapagos Islands	SAN-J	22	0	-47.0	26.0	-42.0
140	South American 1969 - Guyana	SAN-G	22	0	-53.0	3.0	-47.0
141	South American 1969 - Paraguay	SAN-H	22	0	-61.0	2.0	-33.0
142	South American 1969 - Peru	SAN-I	22	0	-58.0	0.0	-44.0
143	South American 1969 - Trinidad & Tobago	SAN-K	22	0	-45.0	12.0	-33.0
144	South American 1969 - Venezuela	SAN-L	22	0	-45.0	8.0	-33.0
145	Zanderij - Suriname	ZAN	20	0	-265.0	120.0	-358.0
146	Antigua Island Astro 1943 - Antigua, Leeward Islands	AIA	7	0	-270.0	13.0	62.0
147	Ascension Island 1958	ASC	20	0	-205.0	107.0	53.0
148	Astro Dos 71/4 - St Helena Island	SHB	20	0	-320.0	550.0	-494.0
149	Bermuda 1957 - Bermuda Islands	BER	6	0	-73.0	213.0	296.0
150	Deception Island, Antarctica	DID	7	0	260.0	12.0	-147.0
151	Fort Thomas 1955 - Nevis, St Kitts, Leeward Islands	FOT	7	0	-7.0	215.0	225.0
152	Graciosa Base SW 1948 - Faial, Graciosa, Pico, Sao Jorge, Terceira Islands (Azores)	GRA	20	0	-104.0	167.0	-38.0
153	ISTS 061 Astro 1968 - South Georgia Islands	ISG	20	0	-794.0	119.0	-298.0
154	L.C. 5 Astro 1961 - Cayman Brac Island	LCF	6	0	42.0	124.0	147.0
155	Montserrat Island Astro 1958 - Montserrat Leeward Islands	ASM	7	0	174.0	359.0	365.0
156	Naparima, BWI - Trinidad & Tobago	NAP	20	0	-10.0	375.0	165.0
157	Observatorio Meteorologico 1939 - Corvo and Flores Islands (Azores)	FLO	20	0	-425.0	-169.0	81.0
158	Pico De Las Nieves - Canary Islands	PLN	20	0	-307.0	-92.0	127.0
159	Porto Santo 1936 - Porto Santo and Madeira Islands	POS	20	0	-499.0	-249.0	314.0
160	Puerto Rico - Puerto Rico & Virgin Islands	PUR	6	0	11.0	72.0	-101.0
161	Qornoq - South Greenland	QUO	20	0	164.0	138.0	-189.0
162	Sao Braz - Soa Miguel, Santa Maria Islands (Azores)	SAO	20	0	-203.0	141.0	53.0
163	Sapper Hill 1943 - East Falkland Island	SAP	20	0	-355.0	21.0	72.0
164	Selvagem Grande 1938 - Salvage Islands	SGM	20	0	-289.0	-124.0	60.0
165	Tristan Astro 1968 - Tristan du Cunha	TDC	20	0	-632.0	438.0	-609.0
166	Anna 1 Astro 1965 - Cocos Islands	ANO	3	0	-491.0	-22.0	435.0
167	Gandajika Base 1970 - Republic of Maldives	GAA	20	0	-133.0	-321.0	50.0
168	ISTS 073 Astro 1969 - Diego Garcia	IST	20	0	208.0	-435.0	-229.0
169	Kerguelen Island 1949 - Kerguelen Island	KEG	20	0	145.0	-187.0	103.0
170	Mahe 1971 - Mahe Island	MIK	7	0	41.0	-220.0	-134.0
171	Reunion - Mascarene Islands	RUE	20	0	94.0	-948.0	-1262.0

Geodetic Datum Defined in Firmware continued

Index	Description	Short	Ellipsoid Index	Rotation, Scale	dX [m]	dY [m]	dZ [m]
172	American Samoa 1962 - American Samoa Islands	AMA	6	0	-115.0	118.0	426.0
173	Astro Beacon E 1945 - Iwo Jima	ATF	20	0	145.0	75.0	-272.0
174	Astro Tern Island (Frig) 1961 - Tern Island	TRN	20	0	114.0	-116.0	-333.0
175	Astronomical Station 1952 - Marcus Island	ASQ	20	0	124.0	-234.0	-25.0
176	Bellevue (IGN) - Efate and Erromango Islands	IBE	20	0	-127.0	-769.0	472.0
177	Canton Astro 1966 - Phoenix Islands	CAO	20	0	298.0	-304.0	-375.0
178	Chatham Island Astro 1971 - Chatham Island (New Zeland)	CHI	20	0	175.0	-38.0	113.0
179	DOS 1968 - Gizo Island (New Georgia Islands)	GIZ	20	0	230.0	-199.0	-752.0
180	Easter Island 1967 - Easter Island	EAS	20	0	211.0	147.0	111.0
181	Geodetic Datum 1949 - New Zealand	GEO	20	0	84.0	-22.0	209.0
182	Guam 1963 - Guam Island	GUA	6	0	-100.0	-248.0	259.0
183	GUX 1 Astro - Guadalcanal Island	DOB	20	0	252.0	-209.0	-751.0
184	Indonesian 1974 - Indonesia	IDN	19	0	-24.0	-15.0	5.0
185	Johnston Island 1961 - Johnston Island	JOH	20	0	189.0	-79.0	-202.0
186	Kusaie Astro 1951 - Caroline Islands, Fed. States of Micronesia	KUS	20	0	647.0	1777.0	-1124.0
187	Luzon - Philippines (excluding Mindanao Island)	LUZ-A	6	0	-133.0	-77.0	-51.0
188	Luzon - Mindanao Island (Philippines)	LUZ-B	6	0	-133.0	-79.0	-72.0
189	Midway Astro 1961 - Midway Islands	MID	20	0	912.0	-58.0	1227.0
190	Old Hawaiian - Mean Solution	OHA-M	6	0	61.0	-285.0	-181.0
191	Old Hawaiian - Hawaii	OHA-A	6	0	89.0	-279.0	-183.0
192	Old Hawaiian - Kauai	OHA-B	6	0	45.0	-290.0	-172.0
193	Old Hawaiian - Maui	OHA-C	6	0	65.0	-290.0	-190.0
194	Old Hawaiian - Oahu	OHA-D	6	0	58.0	-283.0	-182.0
195	Pitcairn Astro 1967 - Pitcairn Island	PIT	20	0	185.0	165.0	42.0
196	Santo (Dos) 1965 - Espirito Santo Island	SAE	20	0	170.0	42.0	84.0
197	Viti Levu 1916 - Viti Levu Island (Fiji Islands)	MVS	7	0	51.0	391.0	-36.0
198	Wake-Eniwetok 1960 - Marshall Islands	ENW	18	0	102.0	52.0	-38.0
199	Wake Island Astro 1952 - Wake Atoll	WAK	20	0	276.0	-57.0	149.0
200	Bukit Rimpah - Bangka and Belitung Islands (Indonesia)	BUR	5	0	-384.0	664.0	-48.0
201	Camp Area Astro - Camp McMurdo Area, Antarctica	CAZ	20	0	-104.0	-129.0	239.0
202	European 1950 - Iraq, Israel, Jordan, Kuwait, Lebanon, Saudi Arabia & Syria	EUR-S	20	0	-103.0	-106.0	-141.0
203	Gunung Segara - Kalimantan (Indonesia)	GSE	5	0	-403.0	684.0	41.0
204	Herat North - Afghanistan	HEN	20	0	-333.0	-222.0	114.0
205	Indian - Pakistan	IND-P	9	0	283.0	682.0	231.0
206	Pulkovo 1942 - Russia	PUK	21	0	28.0	-130.0	-95.0
207	Tananarive Observatory 1925 - Madagascar	TAN	20	0	-189.0	-242.0	-91.0
208	Yacare - Uruguay	YAC	20	0	-155.0	171.0	37.0
209	Krassovsky 1942 - Russia	KRA42	21	0	26.0	-139.0	-80.0
210	Lommel Datum 1950 - Belgium & Luxembourg	BLG50	20	0	-55.0	49.0	-158.0

Geodetic Datum Defined in Firmware continued

Index	*Description*	*Short*	*Ellipsoid Index*	*Rotation, Scale*	*dX [m]*	*dY [m]*	*dZ [m]*
211	Reseau National Belge 1972 - Belgium	RNB72	20	0	-104.0	80.0	-75.0
212	NTF - Nouvelle Triangulation de la France	NTF	7	0	-168.0	-60.0	320.0
213	Netherlands 1921 - Netherlands	NL21	5	0	719.0	47.0	640.0
214	European Datum 1987, IAG RETrig Subcommision.	ED87	20	2	-82.5	-91.7	-117.7
215	Swiss Datum 1903+ (LV95)	CH95	5	0	674.374	15.056	405.346

C.2 Ellipsoids

Ellipsoids

Index	*Description*	*Semi Major Axis [m]*	*Flattening*
0	WGS 84	6378137.000	298.257223563
1	Airy 1830	6377563.396	299.3249646
2	Modified Airy	6377340.189	299.3249646
3	Australian National	6378160.000	298.25
4	Bessel 1841 (Namibia)	6377483.865	299.1528128
5	Bessel 1841	6377397.155	299.1528128
6	Clarke 1866	6378206.400	294.9786982
7	Clarke 1880	6378249.145	293.465
8	Earth-90	6378136.000	298.257839303
9	Everest (India 1830)	6377276.345	300.8017
10	Everest (Sabah Sarawak)	6377298.556	300.8017
11	Everest (India 1956)	6377301.243	300.8017
12	Everest (Malaysia 1969)	6377295.664	300.8017
13	Everest (Malay. & Singapore 1948)	6377304.063	300.8017
14	Everest (Pakistan)	6377309.613	300.8017
15	Modified Fischer 1960	6378155.000	298.3
16	GRS 80	6378137.000	298.257222101
17	Helmert 1906	6378200.000	298.3
18	Hough 1960	6378270.000	297.0
19	Indonesian 1974	6378160.000	298.247
20	International 1924	6378388.000	297.0
21	Krassovsky 1940	6378245.000	298.3
22	South American 1969	6378160.000	298.25
23	WGS 72	6378135.000	298.26

C.3 Rotation and Scale

Rotation and Scale

Index	*Description*	*Rot X [seconds]*	*Rot Y [seconds]*	*Rot Z [seconds]*	*Scale*
0		+0.0000	+0.0000	+0.0000	0.000
1	WGS 72	+0.0000	+0.0000	-0.5540	0.220
2	European Datum 1987 IAG RETrig Subcommision.	+0.1338	-0.0625	-0.0470	0.045

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 Protocol Specification should be used together with the Data Sheet and Hardware Integration Manual of the GPS receiver.

Related Documents for Modules

Documentation for the following products can be downloaded from our website. For other products please contact u-blox.

u-blox 6

- LEA-6 Data Sheet, Docu. No GPS.G6-HW-09004
- NEO-6 Data Sheet, Docu. No GPS.G6-HW-09005
- LEA-6/NEO-6 Hardware Integration Manual, Docu. No GPS.G6-HW-09007

Contact

For complete contact information visit us at www.u-blox.com

Headquarters

u-blox AG
Zuercherstrasse 68
CH-8800 Thalwil
Switzerland

Phone: +41 44 722 74 44
Fax: +41 44 722 74 47
E-mail: info@u-blox.com

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

Europe, Middle East, Africa

u-blox AG
Phone: +41 44 722 74 44
E-mail: info@u-blox.com

Technical Support:
Phone: +41 44 722 74 44
E-mail: info@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 China:
Phone: +86 10 68 133 545
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office Japan:
Phone: +81 03 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