

u-blox M8

Receiver Description

Including Protocol Specification V15

Abstract

The Receiver Description Including Protocol Specification describes the firmware features, specifications and configuration for u-blox M8 high performance high performance positioning modules.

The Receiver Description provides an overview and conceptual details of the supported features. The Protocol Specification details version 15 of the NMEA and UBX protocols and serves as a reference manual.

Document Information

Title	u-blox M8 Receiver Description
Subtitle	Including Protocol Specification V15
Document type	Manual
Document number	UBX-13003221-R01 74408, 3 Oct 2013
Document status	Protocol version 15.00 (Early Production Information)

This document and the use of any information contained therein, is subject to the acceptance of the u-blox terms and conditions. They can be downloaded from www.u-blox.com. u-blox makes no warranties based on the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. u-blox reserves all rights to this document and the information contained herein. Reproduction, use or disclosure to third parties without express permission is strictly prohibited. Copyright © 2013, u-blox AG.

Table of Contents

Receiver Description	1
1 Overview	1
2 Navigation Configuration Settings Description.....	1
2.1 Platform settings.....	1
2.2 Navigation Input Filters	2
2.3 Navigation Output Filters.....	2
2.3.1 Speed (3-D) Low-pass Filter	3
2.3.2 Course over Ground Low-pass Filter	3
2.4 Static Hold	3
2.5 Low-speed Course Over Ground Filtering	3
2.6 Freezing the Course Over Ground	4
2.7 Degraded Navigation.....	4
2.7.1 2D Navigation.....	4
3 Concurrent GNSS.....	4
3.1 Navigation Systems	4
3.1.1 GPS.....	4
3.1.2 GLONASS.....	4
3.1.3 BeiDou	5
3.1.4 SBAS.....	5
3.1.5 QZSS.....	5
3.2 Configuration	5
4 SBAS Configuration Settings Description	5
4.1 SBAS (Satellite Based Augmentation Systems).....	5
4.2 SBAS Features	7
4.3 SBAS Configuration.....	8
5 Clocks and Time.....	8
5.1 Receiver Local Time.....	8
5.2 Navigation Epochs.....	9
5.3 iTOW Timestamps	10
5.4 UTC Representation	10
5.5 Leap Seconds	10
5.6 Real Time Clock	11
6 Serial Communication Ports Description	11
6.1 TX-ready indication.....	11
6.2 Extended TX timeout.....	12
6.3 UART Ports.....	12
6.4 USB Port	12
6.5 DDC Port	13
6.5.1 Read Access.....	13
6.5.1.1 Read Access Forms	14

6.5.2	Write Access.....	15
6.6	SPI Port.....	16
6.6.1	Maximum SPI clock speed.....	16
6.6.2	Read Access.....	16
6.6.3	Back-To-Back Read and Write Access.....	16
6.7	How to change between protocols.....	17
7	Receiver Configuration	17
7.1	Configuration Concept	17
7.2	Organization of the Configuration Sections	18
7.3	Permanent Configuration Storage Media	19
7.4	Receiver Default Configuration	19
8	Forcing a Receiver Reset	19
9	Remote Inventory	20
9.1	Description.....	20
9.2	Usage	20
10	Power Management	20
10.1	Continuous Mode.....	20
10.2	Power Save Mode.....	21
10.2.1	Operation	21
10.2.1.1	ON/OFF operation - long update period	22
10.2.1.2	Cyclic tracking operation - short update period	23
10.2.1.3	User controlled operation - update and search period of zero	23
10.2.1.4	Satellite data download	23
10.2.2	Configuration	24
10.2.2.1	Mode of operation	24
10.2.2.2	Update and search period	24
10.2.2.3	Acquisition timeout	25
10.2.2.4	On time and wait for timefix	25
10.2.2.5	Do not enter 'inactive for search' state when no fix	25
10.2.2.6	Update RTC and Ephemeris.....	25
10.2.2.7	EXTINT pin control	25
10.2.2.8	Grid offset	25
10.2.3	Features	26
10.2.3.1	Communication.....	26
10.2.3.2	Wake-up	26
10.2.3.3	Behavior while USB host connected	26
10.2.3.4	Cooperation with the AssistNow Autonomous feature	26
10.2.4	Examples.....	27
10.2.4.1	Use Grid Offset.....	27
10.2.4.2	Use update periods of zero	27
10.3	Peak current settings	27
10.4	Power On/Off command.....	27
10.5	EXTINT pin control when Power Save Mode is not active.....	27
10.6	Measurement and navigation rate with Power Save Mode.....	27

11	Time pulse	28
11.1	Introduction	28
11.2	Recommendations	28
11.3	Time pulse configuration	29
11.4	Configuring time pulse with UBX-CFG-TP5	29
11.4.1	Example 1:	30
11.4.2	Example 2:	31
12	Receiver Status Monitoring	32
12.1	Input/Output system	32
12.2	Jamming/Interference Indicator	33
12.3	Jamming/Interference Monitor (ITFM)	33
13	Timemark	34
14	Multi-GNSS Assistance (MGA)	35
14.1	Introduction	35
14.2	Assistance Data	35
14.3	AssistNow Online	36
14.3.1	Host Software	37
14.3.2	AssistNow Online Sequence	37
14.3.3	Flow Control	38
14.3.4	Authorization	38
14.3.5	Service Parameters	38
14.3.5.1	Position parameters (lat, lon, alt and pacc)	39
14.3.5.2	Time parameters (tacc and latency)	39
14.3.6	Multiple Servers	40
14.4	AssistNow Offline	40
14.4.1	Service Parameters	41
14.4.2	Authorization	41
14.4.3	Multiple Servers	41
14.4.4	Time, Position and Almanac	41
14.4.5	Flash-based AssistNow Offline	42
14.4.5.1	Flash-based Storage Procedure	42
14.4.6	Host-based AssistNow Offline	43
14.4.6.1	Host-based Procedure	43
14.5	Preserving Information During Power-off	43
14.6	AssistNow Autonomous	44
14.6.1	Introduction	44
14.6.2	Concept	44
14.6.3	Interface	45
14.6.4	Benefits and Drawbacks	46
15	Odometer	47
15.1	Introduction	47
15.2	Odometer Output	47
15.3	Odometer Configuration	47
15.4	Resetting the Odometer	48

16 Logging	48
16.1 Introduction.....	48
16.2 Setting the logging system up	48
16.3 Information about the log	49
16.4 Recording.....	50
16.5 Retrieval.....	51
16.6 Command message acknowledgement	51
NMEA Protocol	53
17 Protocol Overview	53
17.1 Message Format	53
17.2 Talker ID.....	53
17.3 Protocol Configuration	54
17.4 Satellite Numbering	55
17.5 Latitude and Longitude Format	55
17.6 Position Fix Flags	55
17.7 Multi-GNSS considerations	56
17.8 Output of Invalid/Unknown Data	57
17.9 Messages Overview	57
18 Standard Messages	59
18.1 DTM.....	59
18.1.1 Datum Reference	59
18.2 GBQ.....	60
18.2.1 Poll a standard message (if the current Talker ID is GB).....	60
18.3 GBS.....	60
18.3.1 GNSS Satellite Fault Detection	60
18.4 GGA.....	61
18.4.1 Global positioning system fix data.....	61
18.5 GLL	62
18.5.1 Latitude and longitude, with time of position fix and status.....	62
18.6 GLQ	63
18.6.1 Poll a standard message (if the current Talker ID is GL)	63
18.7 GNQ.....	64
18.7.1 Poll a standard message (if the current Talker ID is GN)	64
18.8 GNS.....	64
18.8.1 GNSS fix data	64
18.9 GPQ	65
18.9.1 Poll a standard message (if the current Talker ID is GP).....	65
18.10 GRS.....	66
18.10.1 GNSS Range Residuals.....	66
18.11 GSA.....	67
18.11.1 GNSS DOP and Active Satellites.....	67
18.12 GST	68
18.12.1 GNSS Pseudo Range Error Statistics	68
18.13 GSV.....	69

18.13.1	GNSS Satellites in View	69
18.14	RMC.....	70
18.14.1	Recommended Minimum data.....	70
18.15	TXT	71
18.15.1	Text Transmission	71
18.16	VLW.....	72
18.16.1	Dual ground/water distance.....	72
18.17	VTG.....	72
18.17.1	Course over ground and Ground speed	72
18.18	ZDA	73
18.18.1	Time and Date	73
19	PUBX Messages	75
19.1	CONFIG (PUBX,41)	75
19.1.1	Set Protocols and Baudrate	75
19.2	POSITION (PUBX,00)	76
19.2.1	Lat/Long Position Data.....	76
19.3	SVSTATUS (PUBX,03)	77
19.3.1	Satellite Status.....	77
19.4	TIME (PUBX,04).....	78
19.4.1	Time of Day and Clock Information.....	78
UBX Protocol.....		80
20	UBX Protocol Key Features.....	80
21	UBX Packet Structure.....	80
22	UBX Payload Definition Rules	80
22.1	Structure Packing	80
22.2	Message Naming.....	80
22.3	Number Formats.....	81
23	UBX Checksum.....	81
24	UBX Message Flow.....	82
24.1	Acknowledgement.....	82
24.2	Polling Mechanism	82
25	UBX Satellite Numbering.....	82
26	UBX Class IDs.....	83
27	UBX Messages Overview.....	84
28	UBX-ACK (0x05).....	89
28.1	UBX-ACK-ACK (0x05 0x01).....	89
28.1.1	Message Acknowledged	89
28.2	UBX-ACK-NAK (0x05 0x00)	89
28.2.1	Message Not-Acknowledged.....	89
29	UBX-AID (0x0B).....	90
29.1	UBX-AID-ALM (0x0B 0x30).....	90
29.1.1	Poll GPS Aiding Almanac Data	90
29.1.2	Poll GPS Aiding Almanac Data for a SV	90
29.1.3	GPS Aiding Almanac Input/Output Message.....	91

29.2	UBX-AID-AOP (0x0B 0x33)	91
29.2.1	Poll AssistNow Autonomous data, all GPS satellites	91
29.2.2	Poll AssistNow Autonomous data, one GPS satellite	92
29.2.3	AssistNow Autonomous data	92
29.3	UBX-AID-EPH (0x0B 0x31)	93
29.3.1	Poll GPS Aiding Ephemeris Data	93
29.3.2	Poll GPS Aiding Ephemeris Data for a SV	93
29.3.3	GPS Aiding Ephemeris Input/Output Message	94
29.4	UBX-AID-HUI (0x0B 0x02)	95
29.4.1	Poll GPS Health, UTC and ionosphere parameters	95
29.4.2	GPS Health, UTC and ionosphere parameters	95
29.5	UBX-AID-INI (0x0B 0x01)	96
29.5.1	Poll GPS Initial Aiding Data	96
29.5.2	Aiding position, time, frequency, clock drift	97
30	UBX-CFG (0x06)	99
30.1	UBX-CFG-ANT (0x06 0x13)	99
30.1.1	Poll Antenna Control Settings	99
30.1.2	Antenna Control Settings	99
30.2	UBX-CFG-CFG (0x06 0x09)	100
30.2.1	Clear, Save and Load configurations	100
30.3	UBX-CFG-DAT (0x06 0x06)	102
30.3.1	Poll Datum Setting	102
30.3.2	Set User-defined Datum	102
30.3.3	The currently defined Datum	103
30.4	UBX-CFG-GNSS (0x06 0x3E)	104
30.4.1	Poll the GNSS system configuration	104
30.4.2	GNSS system configuration	104
30.5	UBX-CFG-INF (0x06 0x02)	105
30.5.1	Poll INF message configuration for one protocol	105
30.5.2	Information message configuration	106
30.6	UBX-CFG-ITFM (0x06 0x39)	107
30.6.1	Poll Jamming/Interference Monitor configuration	107
30.6.2	Jamming/Interference Monitor configuration	107
30.7	UBX-CFG-LOGFILTER (0x06 0x47)	108
30.7.1	Poll Data Logger filter Configuration	108
30.7.2	Data Logger Configuration	108
30.8	UBX-CFG-MSG (0x06 0x01)	110
30.8.1	Poll a message configuration	110
30.8.2	Set Message Rate(s)	110
30.8.3	Set Message Rate	111
30.9	UBX-CFG-NAV5 (0x06 0x24)	111
30.9.1	Poll Navigation Engine Settings	111
30.9.2	Navigation Engine Settings	111
30.10	UBX-CFG-NAVX5 (0x06 0x23)	113

30.10.1	Poll Navigation Engine Expert Settings	113
30.10.2	Navigation Engine Expert Settings	113
30.11	UBX-CFG-NMEA (0x06 0x17)	115
30.11.1	Poll the NMEA protocol configuration	115
30.11.2	NMEA protocol configuration (deprecated)	115
30.11.3	NMEA protocol configuration V0 (deprecated)	117
30.11.4	Extended NMEA protocol configuration V1	119
30.12	UBX-CFG-ODO (0x06 0x1E)	122
30.12.1	Poll Odometer, Low-speed COG Engine Settings	122
30.12.2	Odometer, Low-speed COG Engine Settings	122
30.13	UBX-CFG-PM2 (0x06 0x3B)	123
30.13.1	Poll extended Power Management configuration	123
30.13.2	Extended Power Management configuration	124
30.14	UBX-CFG-PRT (0x06 0x00)	125
30.14.1	Polls the configuration of the used I/O Port	125
30.14.2	Polls the configuration for one I/O Port	126
30.14.3	Port Configuration for UART	126
30.14.4	Port Configuration for USB Port	129
30.14.5	Port Configuration for SPI Port	131
30.14.6	Port Configuration for DDC Port	133
30.15	UBX-CFG-PWR (0x06 0x57)	136
30.15.1	Put the receiver in a defined power state	136
30.16	UBX-CFG-RATE (0x06 0x08)	136
30.16.1	Poll Navigation/Measurement Rate Settings	136
30.16.2	Navigation/Measurement Rate Settings	137
30.17	UBX-CFG-RINV (0x06 0x34)	137
30.17.1	Poll contents of Remote Inventory	137
30.17.2	Contents of Remote Inventory	138
30.18	UBX-CFG-RST (0x06 0x04)	138
30.18.1	Reset Receiver / Clear Backup Data Structures	138
30.19	UBX-CFG-RXM (0x06 0x11)	140
30.19.1	Poll RXM configuration	140
30.20	UBX-CFG-SBAS (0x06 0x16)	140
30.20.1	Poll contents of SBAS Configuration	140
30.20.2	SBAS Configuration	140
30.21	UBX-CFG-TP5 (0x06 0x31)	142
30.21.1	Poll Time Pulse Parameters	142
30.21.2	Poll Time Pulse Parameters	142
30.21.3	Time Pulse Parameters	143
30.22	UBX-CFG-USB (0x06 0x1B)	144
30.22.1	Poll a USB configuration	144
30.22.2	USB Configuration	144
31	UBX-INF (0x04)	146
31.1	UBX-INF-DEBUG (0x04 0x04)	146

31.1.1	ASCII String output, indicating debug output	146
31.2	UBX-INF-ERROR (0x04 0x00)	146
31.2.1	ASCII String output, indicating an error	146
31.3	UBX-INF-NOTICE (0x04 0x02)	147
31.3.1	ASCII String output, with informational contents	147
31.4	UBX-INF-TEST (0x04 0x03)	147
31.4.1	ASCII String output, indicating test output	147
31.5	UBX-INF-WARNING (0x04 0x01)	148
31.5.1	ASCII String output, indicating a warning	148
32	UBX-LOG (0x21)	149
32.1	UBX-LOG-CREATE (0x21 0x07)	149
32.1.1	Create Log File	149
32.2	UBX-LOG-ERASE (0x21 0x03)	150
32.2.1	Erase Logged Data	150
32.3	UBX-LOG-FINDTIME (0x21 0x0E)	150
32.3.1	Find index of the first log entry <= given time	150
32.3.2	Response to FINDTIME request.	151
32.4	UBX-LOG-INFO (0x21 0x08)	151
32.4.1	Poll for log information	151
32.4.2	Log information	151
32.5	UBX-LOG-RETRIEVEPOSEXTRA (0x21 0x0f)	153
32.5.1	Odometer log entry	153
32.6	UBX-LOG-RETRIEVEPOS (0x21 0x0b)	154
32.6.1	Position fix log entry	154
32.7	UBX-LOG-RETRIEVESTRING (0x21 0x0d)	155
32.7.1	Byte string log entry	155
32.8	UBX-LOG-RETRIEVE (0x21 0x09)	155
32.8.1	Request log data	155
32.9	UBX-LOG-STRING (0x21 0x04)	156
32.9.1	Store arbitrary string in on-board Flash memory	156
33	UBX-MGA (0x13)	157
33.1	UBX-MGA-ACK (0x13 0x60)	157
33.1.1	UBX-MGA-ACK-DATA0	157
33.2	UBX-MGA-ANO (0x13 0x20)	158
33.2.1	Multi-GNSS AssistNow Offline Assistance	158
33.3	UBX-MGA-DBD (0x13 0x80)	158
33.3.1	Poll the Navigation Database	158
33.3.2	Navigation Database Dump Entry	159
33.4	UBX-MGA-FLASH (0x13 0x21)	159
33.4.1	UBX-MGA-FLASH-DATA	159
33.4.2	UBX-MGA-FLASH-STOP	160
33.4.3	UBX-MGA-FLASH-ACK	160
33.5	UBX-MGA-GLO (0x13 0x06)	161
33.5.1	UBX-MGA-GLO-EPH	161

33.5.2	UBX-MGA-GLO-ALM	162
33.5.3	UBX-MGA-GLO-TIMEOFFSET	163
33.6	UBX-MGA-GPS (0x13 0x00)	164
33.6.1	UBX-MGA-GPS-EPH	164
33.6.2	UBX-MGA-GPS-ALM	165
33.6.3	UBX-MGA-GPS-HEALTH	166
33.6.4	UBX-MGA-GPS-UTC	167
33.6.5	UBX-MGA-GPS-IONO	167
33.7	UBX-MGA-INI (0x13 0x40)	168
33.7.1	UBX-MGA-INI-POS_XYZ	168
33.7.2	UBX-MGA-INI-POS_LLH	169
33.7.3	UBX-MGA-INI-TIME.UTC	170
33.7.4	UBX-MGA-INI-TIME_GNSS	171
33.7.5	UBX-MGA-INI-CLKD	172
33.7.6	UBX-MGA-INI-FREQ	173
33.7.7	UBX-MGA-INI-EOP	174
33.8	UBX-MGA-QZSS (0x13 0x05)	174
33.8.1	UBX-MGA-QZSS-EPH	174
33.8.2	UBX-MGA-QZSS-ALM	176
33.8.3	UBX-MGA-QZSS-HEALTH	177
34	UBX-MON (0x0A)	178
34.1	UBX-MON-GNSS (0x0A 0x28)	178
34.1.1	Information message GNSS selection	178
34.2	UBX-MON-HW2 (0x0A 0x0B)	179
34.2.1	Extended Hardware Status	179
34.3	UBX-MON-HW (0x0A 0x09)	180
34.3.1	Hardware Status	180
34.4	UBX-MON-IO (0x0A 0x02)	182
34.4.1	I/O Subsystem Status	182
34.5	UBX-MON-MSGPP (0x0A 0x06)	182
34.5.1	Message Parse and Process Status	182
34.6	UBX-MON-PATCH (0x0A 0x27)	183
34.6.1	Poll Request for installed patches	183
34.6.2	Output information about installed patches	183
34.7	UBX-MON-RXBUF (0x0A 0x07)	184
34.7.1	Receiver Buffer Status	184
34.8	UBX-MON-RXR (0x0A 0x21)	185
34.8.1	Receiver Status Information	185
34.9	UBX-MON-TXBUF (0x0A 0x08)	185
34.9.1	Transmitter Buffer Status	185
34.10	UBX-MON-VER (0x0A 0x04)	186
34.10.1	Poll Receiver/Software Version	186
34.10.2	Receiver/Software Version	187
35	UBX-NAV (0x01)	188

35.1	UBX-NAV-AOPSTATUS (0x01 0x60)	188
35.1.1	AssistNow Autonomous Status	188
35.2	UBX-NAV-CLOCK (0x01 0x22)	189
35.2.1	Clock Solution	189
35.3	UBX-NAV-DGPS (0x01 0x31)	189
35.3.1	DGPS Data Used for NAV	189
35.4	UBX-NAV-DOP (0x01 0x04)	190
35.4.1	Dilution of precision	190
35.5	UBX-NAV-ODO (0x01 0x09)	191
35.5.1	Odometer Solution	191
35.6	UBX-NAV-ORB (0x01 0x34)	191
35.6.1	GNSS Orbit Database Info	191
35.7	UBX-NAV-POSECEF (0x01 0x01)	194
35.7.1	Position Solution in ECEF	194
35.8	UBX-NAV-POSLLH (0x01 0x02)	195
35.8.1	Geodetic Position Solution	195
35.9	UBX-NAV-PVT (0x01 0x07)	195
35.9.1	Navigation Position Velocity Time Solution	195
35.10	UBX-NAV-RESETODO (0x01 0x10)	198
35.10.1	Reset odometer	198
35.11	UBX-NAV-SAT (0x01 0x35)	198
35.11.1	Satellite Information	198
35.12	UBX-NAV-SBAS (0x01 0x32)	200
35.12.1	SBAS Status Data	200
35.13	UBX-NAV-SOL (0x01 0x06)	201
35.13.1	Navigation Solution Information	201
35.14	UBX-NAV-STATUS (0x01 0x03)	202
35.14.1	Receiver Navigation Status	202
35.15	UBX-NAV-SVINFO (0x01 0x30)	204
35.15.1	Space Vehicle Information	204
35.16	UBX-NAV-TIMEGPS (0x01 0x20)	206
35.16.1	GPS Time Solution	206
35.17	UBX-NAV-TIMEUTC (0x01 0x21)	207
35.17.1	UTC Time Solution	207
35.18	UBX-NAV-VELECEF (0x01 0x11)	208
35.18.1	Velocity Solution in ECEF	208
35.19	UBX-NAV-VELNED (0x01 0x12)	209
35.19.1	Velocity Solution in NED	209
36	UBX-RXM (0x02)	210
36.1	UBX-RXM-PMREQ (0x02 0x41)	210
36.1.1	Requests a Power Management task	210
36.2	UBX-RXM-SVSI (0x02 0x20)	210
36.2.1	SV Status Info	210
37	UBX-TIM (0x0D)	213

37.1	UBX-TIM-TM2 (0x0D 0x03).....	213
37.1.1	Time mark data	213
37.2	UBX-TIM-TP (0x0D 0x01)	214
37.2.1	Time Pulse Timedata	214
37.3	UBX-TIM-VRFY (0x0D 0x06)	215
37.3.1	Sourced Time Verification	215
38	UBX-UPD (0x09).....	217
38.1	UBX-UPD-SOS (0x09 0x14)	217
38.1.1	Poll Backup File Restore Status.....	217
38.1.2	Create Backup File in Flash	217
38.1.3	Clear Backup in Flash	218
38.1.4	Backup File Creation Acknowledge	218
38.1.5	System Restored from Backup.....	219
	RTCM Protocol	220
39	Introduction.....	220
40	Supported Messages.....	220
41	Configuration	220
42	Output	220
43	Restrictions	221
44	Reference	221
	Appendix	222
A	Protocol Versions.....	222
A.1	Supported Protocol Versions	222
B	Satellite Numbering.....	222
C	u-blox M8 Default Settings	223
C.1	Antenna Supervisor Settings (UBX-CFG-ANT)	223
C.2	Datum Settings (UBX-CFG-DAT)	223
C.3	Navigation Settings (UBX-CFG-NAV5)	224
C.4	Navigation Settings (UBX-CFG-NAVX5)	224
C.5	Output Rates (UBX-CFG-RATE).....	224
C.6	Power Management 2 Configuration (UBX-CFG-PM2).....	225
C.7	Receiver Manager Configuration (UBX-CFG-RXM)	225
C.8	GNSS system configuration (UBX-CFG-GNSS)	225
C.9	SBAS Configuration (UBX-CFG-SBAS)	225
C.10	Port Configuration (UBX-CFG-PRT)	226
C.10.1	UART Port Configuration	226
C.10.2	USB Port Configuration	226
C.10.3	SPI Port Configuration	226
C.10.4	DDC Port Configuration	227
C.11	USB Settings (UBX-CFG-USB)	227
C.12	Message Settings (UBX-CFG-MSG)	227
C.13	NMEA Protocol Settings (UBX-CFG-NMEA)	227
C.14	Logging Configuration (UBX-CFG-LOGFILTER)	228
C.15	Remote Inventory (UBX-CFG-RINV).....	228

C.16	INF Messages Settings (UBX-CFG-INF)	229
C.17	Timepulse Settings (UBX-CFG-TP5)	229
C.18	Jammer/Interference Monitor (UBX-CFG-ITFM)	230
D	u-blox M8 Standard firmware versions	230
Related Documents		231
	Overview	231
Revision History		232
Contact.....		233
	u-blox Offices	233

Receiver Description

1 Overview

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

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

The Protocol Specification is a reference describing the software messages used by your u-blox GNSS (Global Navigation Satellite System: e.g. GPS, GLONASS, etc.) receiver and is organized by the specific NMEA and UBX messages.



This document provides general information on u-blox GNSS receivers. Some information might not apply to certain products. Refer to the product Data Sheet and/or Hardware Integration Manual for possible restrictions or limitations.

2 Navigation Configuration Settings Description

This section relates to the configuration message [UBX-CFG-NAV5](#).

2.1 Platform settings

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

Dynamic Platform Models

Platform	Description
Portable	Applications with low acceleration, e.g. portable devices. Suitable for most situations.
Stationary	Used in timing applications (antenna must be stationary) or other stationary applications. Velocity restricted to 0 m/s. Zero dynamics assumed.
Pedestrian	Applications with low acceleration and speed, e.g. how a pedestrian would move. Low acceleration assumed.
Automotive	Used for applications with equivalent dynamics to those of a passenger car. Low vertical acceleration assumed.
At sea	Recommended for applications at sea, with zero vertical velocity. Zero vertical velocity assumed. Sea level assumed.
Airborne <1g	Used for applications with a higher dynamic range and vertical acceleration than a passenger car. No 2D position fixes supported.
Airborne <2g	Recommended for typical airborne environment. No 2D position fixes supported.
Airborne <4g	Only recommended for extremely dynamic environments. No 2D position fixes supported.

Dynamic Platform Model Details

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



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 only calculate 2D (2D only) or 3D (3D only) positions.
fixedAlt and fixedAltVar	The fixed altitude is used if fixMode is set to 2D only. A variance greater than zero must also be supplied.
minElev	Minimum elevation of a satellite above the horizon in order to be used in the navigation solution. Low elevation satellites may provide degraded accuracy, due to the long signal path through the atmosphere.
cnoThreshNumSVs and cnoThresh	A navigation solution will only be attempted if there are at least the given number of SVs with signals at least as strong as the given threshold.

See also comments in section [Degraded Navigation](#) below.

2.3 Navigation Output Filters

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

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

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



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



The [UBX-NAV-SOL](#) and [UBX-NAV-STATUS](#) messages also report whether a fix is valid in their gpsFixOK and GPSfixOk flags. These messages have only been retained for backwards compatibility

and users are recommended to use the [UBX-NAV-PVT](#) message in preference.

The [UBX-CFG-NAV5](#) message also defines TDOP and time accuracy values that are used in order to establish whether a fix is regarded as locked to GNSS or not, and as a consequence of this, which time pulse setting has to be used. Fixes that do not meet both criteria will be regarded as unlocked to GNSS, and the corresponding time pulse settings of [UBX-CFG-TP5](#) will be used to generate a time pulse.

2.3.1 Speed (3-D) Low-pass Filter

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



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

2.3.2 Course over Ground Low-pass Filter

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



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

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.

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

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

2.5 Low-speed Course Over Ground Filtering

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

2.6 Freezing the Course Over Ground

If the low-speed course over ground filter is deactivated or inactive (see section [Course over Ground Low-speed Filter](#)), the receiver derives the course over ground from the GNSS velocity information. If the velocity cannot be calculated with sufficient accuracy (e.g., with bad signals) or if the absolute speed value is very low (under 0.1 m/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 explicitly configured to do so (see [NMEA Protocol Configuration](#)).

2.7 Degraded Navigation

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

2.7.1 2D Navigation

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



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

3 Concurrent GNSS

The latest products from u-blox are multi-GNSS receivers capable of receiving and processing signals from multiple Global Navigation Satellite Systems (GNSS).

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

3.1 Navigation Systems

This sections briefly describes the different navigation and augmentation systems.

3.1.1 GPS

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



GPS receivers are unaffected by leap second changes as their time base (GPS time) is independent of leap seconds. GPS satellites periodically transmit information that allows the receiver to calculate UTC.

3.1.2 GLONASS

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

It has a number of significant differences when compared to GPS. In most cases, u-blox receivers operate in a very similar manner when they are configured to use GLONASS signals instead of GPS. However some aspects of receiver output are likely to be noticeably affected.

3.1.3 BeiDou

BeiDou is a GNSS operated by China. Its purpose is to initially provide navigation in Asia. In a later stage when the system is fully deployed it will have worldwide coverage. The full system will consist of five geostationary, five inclined geosynchronous and 27 medium earth orbit satellites, as well as control, upload and monitoring stations.

3.1.4 SBAS

There are a number of Space Based Augmentation Systems (SBAS) operated by different countries.

- WAAS (Wide Area Augmentation System) operated by the US.
- EGNOS (European Geostationary Navigation Overlay Service) operated by the EU.
- MSAS (MULTi-functional Satellite Augmentation System) operated by Japan.

See section [SBAS](#) for more details.

3.1.5 QZSS

The Quasi Zenith Satellite System (QZSS) is a regional satellite augmentation system operated by [Japan Aerospace Exploration Agency](#) (JAXA). It is intended as an enhancement to GPS, to increase availability and positional accuracy. The QZSS system achieves this by transmitting GPS-compatible signals in the GPS bands. NMEA messages will show the QZSS satellites only if configured to do so (see section [Satellite Numbering](#)).

3.2 Configuration

Use the [UBX-CFG-GNSS](#) message to configure the u-blox receiver into the required mode of operation. This message allows the user to specify which GNSS signals should be processed along with limits on how many tracking channels should be allocated to each GNSS. The receiver will respond to such a request with a [UBX-ACK-ACK](#) message if it can support the requested configuration or a [UBX-ACK-NAK](#) message if not.

The combinations of systems, which can be configured simultaneously depends on the receivers capability to receive several carrier frequencies. Please check the data sheet of your receiver. Usually GPS, SBAS (e.g. WAAS, EGNOS, MSAS), QZSS L1 and Galileo can be enabled together, because they all use the 1575.42MHz L1 frequency. GLONASS and BeiDou both operate on different frequencies, therefore the receiver must be able to receive a second or even third carrier frequency in order to process these systems together with GPS.



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

4 SBAS Configuration Settings Description

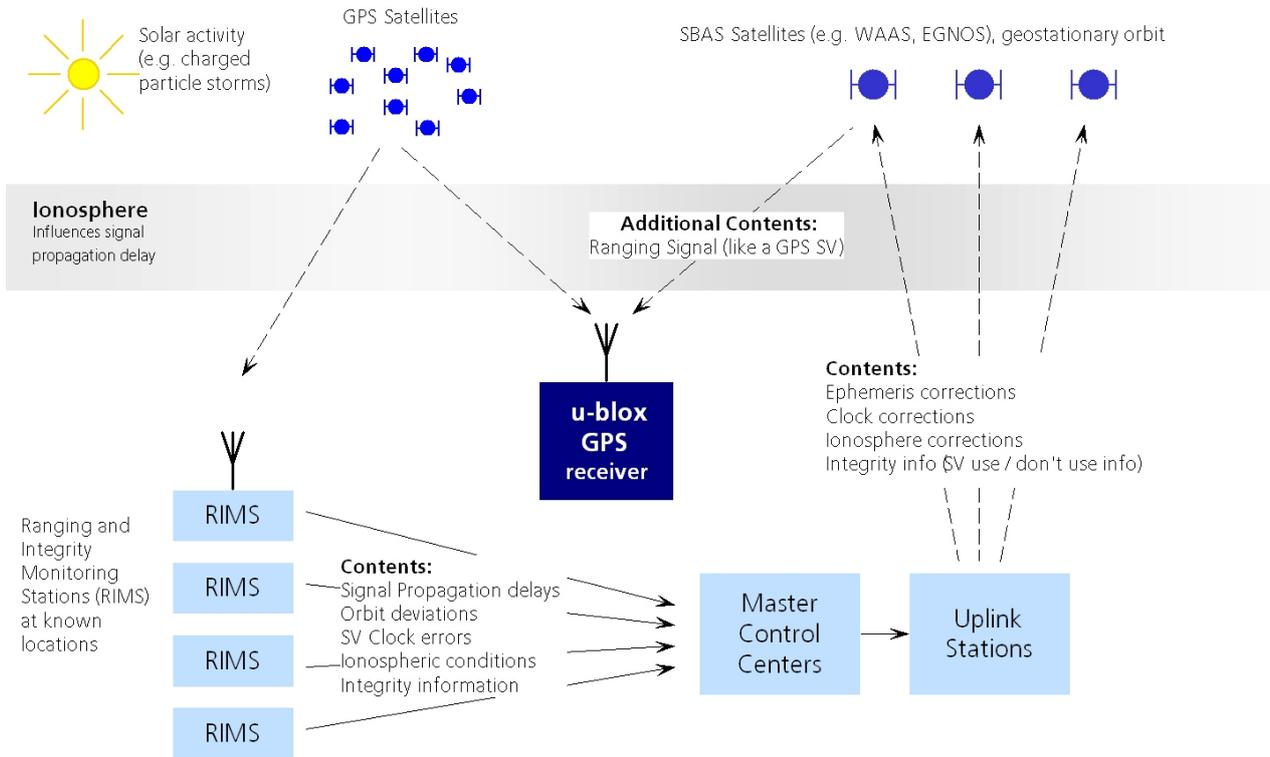
4.1 SBAS (Satellite Based Augmentation Systems)

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



Currently, there are no operational augmentation systems for any GNSS other than GPS. Consequently this section only addresses GPS.

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) has been in operation since 2009.
- GAGAN (GPS Aided Geo Augmented Navigation), developed by the Indian government is at the time of writing in test mode.

Support of SBAS allows u-blox GPS technology to take full advantage of the augmentation systems that are currently available (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, refer to the following resources:

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

- www.isro.org for information on GAGAN.

SBAS satellites tracked (as of June 2013)

Identification	Position	GPS PRN	SBAS Provider
AMR	98° W	133	WAAS
PanAmSat Galaxy XV	133.1° W	135	WAAS
TeleSat Anik F1R	107.3° W	138	WAAS
Inmarsat 3F2 AOR-E	15.5° W	120	EGNOS
Artemis	21.5° W	124	EGNOS
Inmarsat 3F5 IOR-W	25° E	126	EGNOS
MTSAT-1R	140° E	129	MSAS
MTSAT-2	145° E	137	MSAS
GSAT-8	55.1° E	127	GAGAN
GSAT-10	83° E	128	GAGAN

4.2 SBAS Features



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

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

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

If corrections are available from the chosen SBAS satellite and used in the navigation calculation, the DGPS flag is set in the receiver's output protocol messages (see [NAV-PVT](#), [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 regional RIMS stations are combined to make a TEC (Total Electron Content) Map. This map is transferred to the receiver via the satellites to allow a correction of the ionosphere error on each received satellite.

Supported SBAS messages

Message Type	Message Content	Source
0(0/2)	Test Mode	All
1	PRN Mask Assignment	Primary
2, 3, 4, 5	Fast Corrections	Primary
6	Integrity	Primary
7	Fast Correction Degradation	Primary
9	Satellite Navigation (Ephemeris)	All
10	Degradation	Primary
12	Time Offset	Primary
17	Satellite Almanac	All

Supported SBAS messages continued

Message Type	Message Content	Source
18	Ionosphere Grid Point Assignment	Primary
24	Mixed Fast / Long term Corrections	Primary
25	Long term Corrections	Primary
26	Ionosphere Delays	Primary

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

Example 1: SBAS Receiver in North America

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

Example 2: SBAS Receiver in Europe

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



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



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

4.3 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
Services/Usage - Apply integrity information	Use integrity data
Number of tracking channels	Should be set using UBX-CFG-GNSS . The field in UBX-CFG-SBAS is no longer supported.
PRN Mask	Allows selectively enabling/disabling SBAS satellites (e.g. restrict SBAS usage to WAAS-only).

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

5 Clocks and Time

5.1 Receiver Local Time

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

signals is arranged to be synchronised with the "ticking" of this 1 kHz clock signal.

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

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

5.2 Navigation Epochs

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

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

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

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

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



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

5.3 iTOW Timestamps

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

Note that iTOW values may not be valid (i.e. they may have been generated with insufficient conversion data) and therefore it is not recommended to use the iTOW field for any other purpose. If reliable absolute time information is required, users are recommended to use the [UBX-NAV-TIMEUTC](#), [UBX-NAV-TIMEGPS](#), [UBX-NAV-PVT](#) or [UBX-NAV-SOL](#) messages, which contain additional fields that indicate the validity and accuracy of the calculated times.



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

5.4 UTC Representation

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

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

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

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

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

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

Which master clock the UTC time is referenced to is output in the message [UBX-NAV-TIMEUTC](#).

5.5 Leap Seconds

Occasionally it is decided (by one of the international time keeping bodies) that, due to the slightly uneven spin rate of the Earth, UTC has moved sufficiently out of alignment with mean solar time (i.e. the Sun no longer appears directly overhead at 0 longitude at midday). A "leap second" is therefore announced to bring UTC back into close alignment. This normally involves adding an extra second to the last minute of the year, but it

can also happen on 30th June. When this happens UTC clocks are expected to go from 23:59:59 to 23:59:60 and only then on to 00:00:00.

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

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

5.6 Real Time Clock

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

6 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.

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

Port Number assignment

Port #	Electrical Interface
0	DDC (I ² C compatible)
1	UART 1
3	USB
4	SPI

6.1 TX-ready indication

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

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

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

6.2 Extended TX timeout

If the host does not communicate over SPI or DDC for more than approximately 2 seconds, the device assumes that the host is no longer using this interface and no more packets are scheduled for this port. This mechanism can be changed 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.

6.3 UART Ports

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

The serial ports consist of an RX and a TX line. Neither handshaking signals nor hardware flow control signals are available. These serial ports operate in asynchronous mode. The baud rates can be configured individually for each serial port. However, there is no support for setting different baud rates for reception and transmission 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
38400	8	none	1
57600	8	none	1
115200	8	none	1

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

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

See [CFG-PRT for UART](#) for a description of the contents of the UART port configuration message.

6.4 USB Port

One Universal Serial Bus ([USB](#)) port is featured. See the Data Sheet of your specific product for availability. This port can be used for communication purposes and to power the positioning chip or module.

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. See the table below for the default maximum current that can be drawn by the receiver. 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 GNSS functionality when the host suspends the device, e.g. when the host is put into stand-by mode.

Maximum Current in Bus Powered Mode

Generation	Max Current
u-blox M8	100 mA



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

6.5 DDC Port

The Display Data Channel ([DDC](#)) bus is a two-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 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 DDC 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 receiver DDC interface implements a simple streaming interface that allows the constant polling of data, discarding everything that is not parse-able. The receiver returns 0xFF if no data is available. The [TX-ready](#) feature can be used to inform the master about data availability and can be used as a trigger for data transmission.

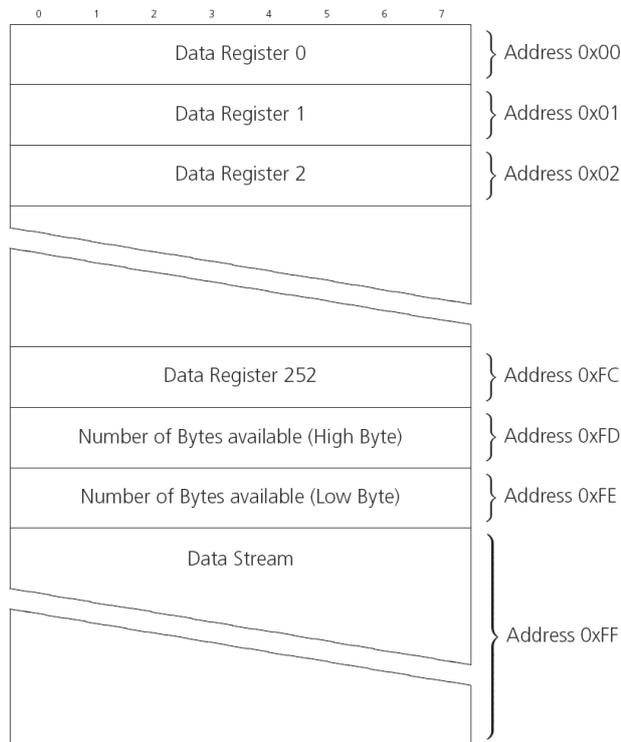
6.5.1 Read Access

The DDC interface allows 256 slave registers to be addressed. As shown in [Figure DDC Register Layout](#) only three of these are currently implemented. The data registers 0 to 252, at addresses 0x00 to 0xFC, each 1 byte in size, contain information to be defined later - the result of reading them is undefined. The currently available number of bytes in the message stream can be read at addresses 0xFD and 0xFE. The register at address 0xFF allows the data stream to be read. If there is no data awaiting transmission from the receiver, then this register will deliver the value 0xff, which cannot be the first byte of a valid message. If message data is ready for transmission then successive reads of register 0xff will deliver the waiting message data.



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!

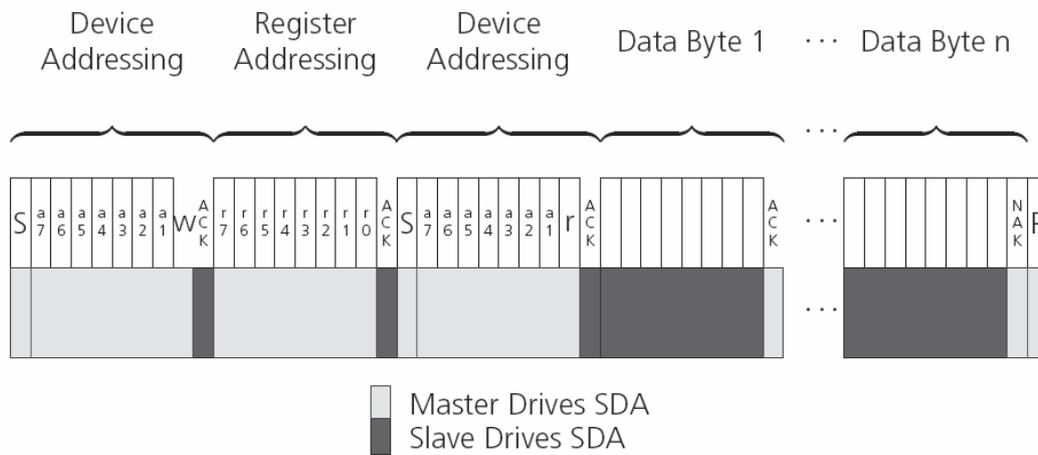
DDC Register Layout



6.5.1.1 Read Access Forms

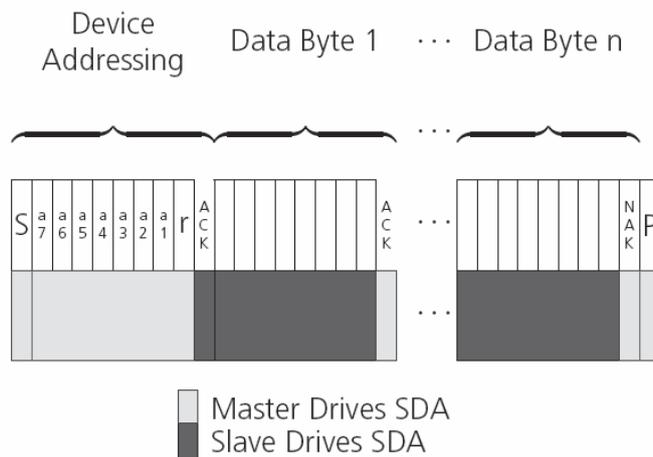
There are two forms of DDC read transfer. The 'random access' form includes a slave register address and thus allows any register to be read. The second 'current address' form omits the register address. If this second form is used then an address pointer in the receiver is used to determine which register to read. This address pointer will increment after each read unless it is already pointing at register 0xff, the highest addressable register, in which case it remains unaltered. The initial value of this address pointer at startup is 0xff, so by default all current address reads will repeatedly read register 0xff and receive the next byte of message data (or 0xff if no message data is waiting). Figure *DDC Random Read Access* shows the format of the random access form of the request. Following the start condition from the master, the 7-bit device address and the \overline{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 recognises the 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 \overline{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.

DDC Random Read Access



The format of the current address read request is :

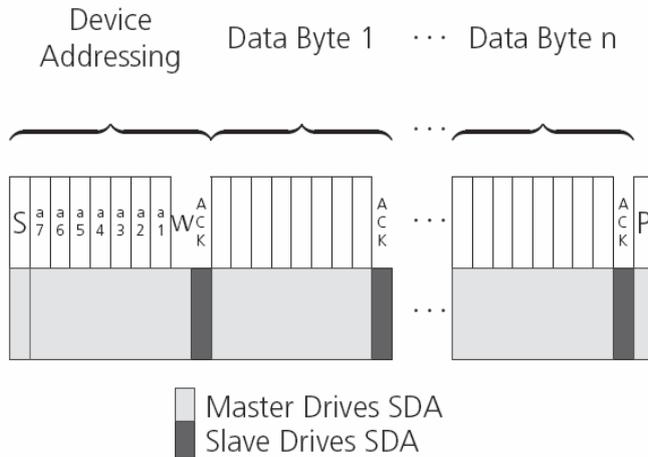
DDC Current Address Read Access



6.5.2 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. 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 $\bar{R}W$ 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



6.6 SPI Port

A Serial Peripheral Interface ([SPI](#)) bus is available with selected receivers. See our online product descriptions 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 `modeSpi` in [CFG-PRT for SPI](#) (default is SPI mode 0).



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

6.6.1 Maximum SPI clock speed

6.6.2 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.

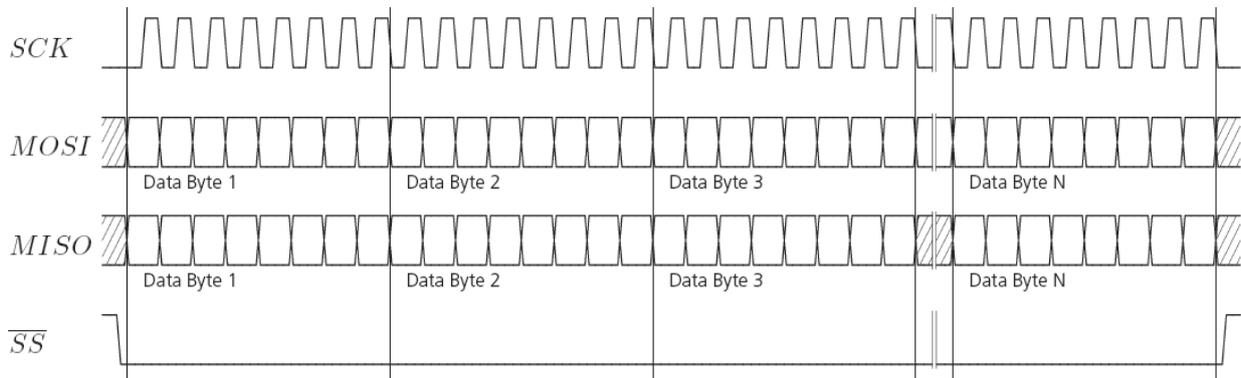
To prevent the receiver from being busy parsing incoming data, the parsing process is stopped after 50 subsequent bytes containing 0xFF. The parsing process is re-enabled with the first byte not equal to 0xFF. The number of bytes to wait for deactivation (50 by default) can be adjusted using the field `waitCnt` 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.

6.6.3 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 simultaneously 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



6.7 How to change between protocols

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

- Step 1: the preferred protocol(s) needs to be enabled on a port using [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.
- Step 2: activate certain messages on each port using [CFG-MSG](#).

7 Receiver Configuration

7.1 Configuration Concept

u-blox positioning technology is fully configurable with UBX protocol configuration messages (message class UBX-CFG). The configuration used by the GNSS 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 GNSS receiver always uses only the Current Configuration.

Unless the Current Configuration is made permanent by using [UBX-CFG-CFG](#) as described below, the Current Configuration will be lost when there is:

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

See the [section on resetting a receiver](#) for details.

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

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

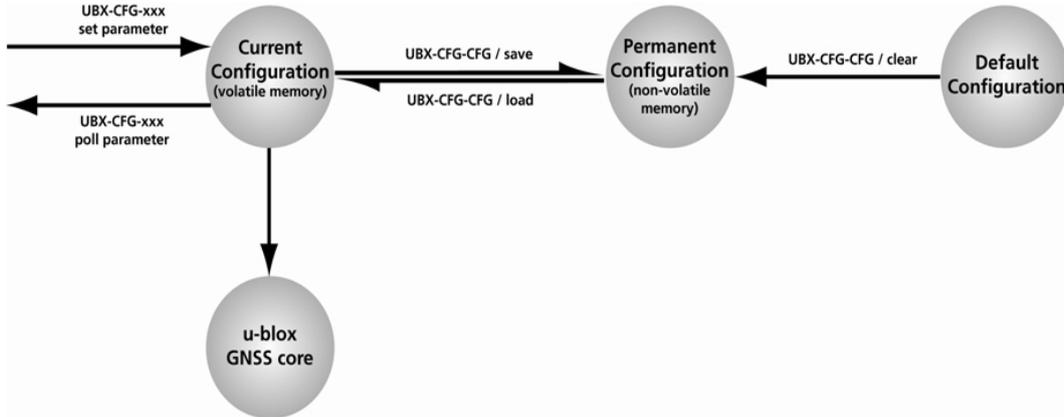
The Permanent Configuration can be restored to the receiver's Default Configuration by sending a [UBX-CFG-CFG](#) message with an appropriate **clearMask** (UBX-CFG-CFG/clear) to the receiver.

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:



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

7.2 Organization of the Configuration Sections

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

Configuration sub-sections

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

Configuration sub-sections continued

Number	Name	CFG messages	Description
10	ANT	UBX-CFG-ANT	Antenna configuration

7.3 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 if any of the reset events listed in the section above occur. By using UBX-CFG-CFG/save, the selected configuration sub-sections are saved to all non-volatile memories available:

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

7.4 Receiver Default Configuration

The Permanent Configuration can be reset to Default Configuration through a [UBX-CFG-CFG/clear](#) message. The receiver's Default Configuration is normally determined when the receiver is manufactured. Refer to specific product data sheet for further details.

8 Forcing a Receiver Reset

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

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

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



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

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

- **Hardware Reset** uses the on-chip Watchdog, in order to electrically reset the chip. This is an immediate, asynchronous reset. No Stop events are generated. This is equivalent to 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 GNSS satellites.
- **Controlled Software Reset (GNSS only)** only restarts the GNSS tasks, without reinitializing the full system or reloading any stored configuration.
- **Controlled GNSS Stop** stops all GNSS tasks. The receiver will not be restarted, but will stop any GNSS related processing.
- **Controlled GNSS Start** starts all GNSS tasks.

9 Remote Inventory

9.1 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.

9.2 Usage

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

Default configuration

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



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

10 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](#) and configured using [UBX-CFG-PM2](#).

10.1 Continuous Mode

During a cold start, a receiver in Continuous Mode continuously deploys the acquisition engine to search for all satellites. 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. Whenever the receiver can no longer calculate a position or the number of satellites tracked is below the sufficient number, the acquisition engine is powered on again to guarantee a quick reacquisition.

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

10.2 Power Save Mode

Power Save Mode (PSM) allows a reduction in system power consumption by selectively switching parts of the receiver on and off.



Note: Power Save Mode cannot be selected when the receiver is configured to process GLONASS signals.

10.2.1 Operation

Power Save Mode has two modes of operation:

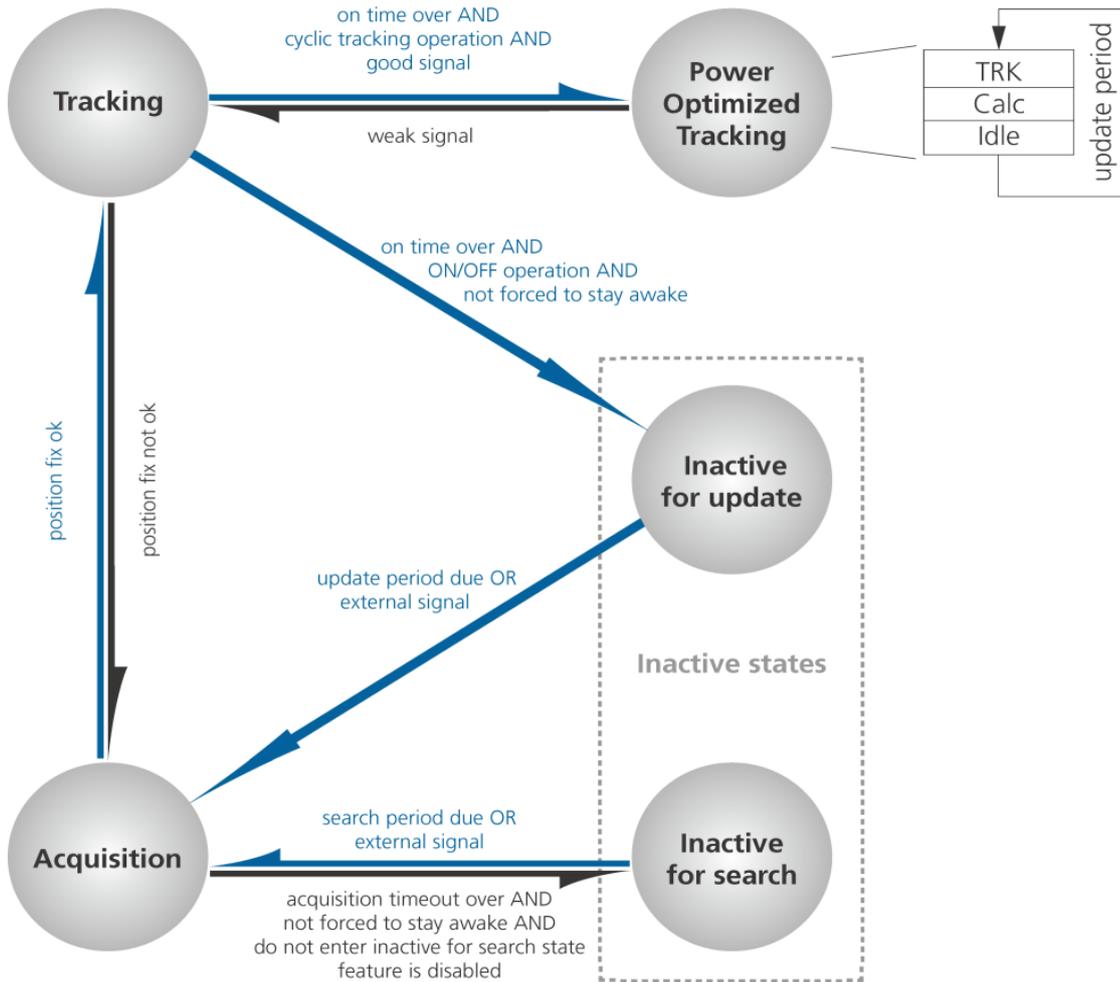
- *Cyclic tracking operation* is used when position fixes are required in short periods of 1 to 10s
- *ON/OFF operation* is used for periods longer than 10s, and can be in the order of minutes, hours or days.

The mode of operation can be configured, and depending on the setting, the receiver demonstrates different behavior: In ON/OFF operation the receiver switches between phases of start-up/navigation and phases with low or almost no system activity. In cyclic tracking the receiver does not shut down completely between fixes, but uses low power tracking instead.

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.

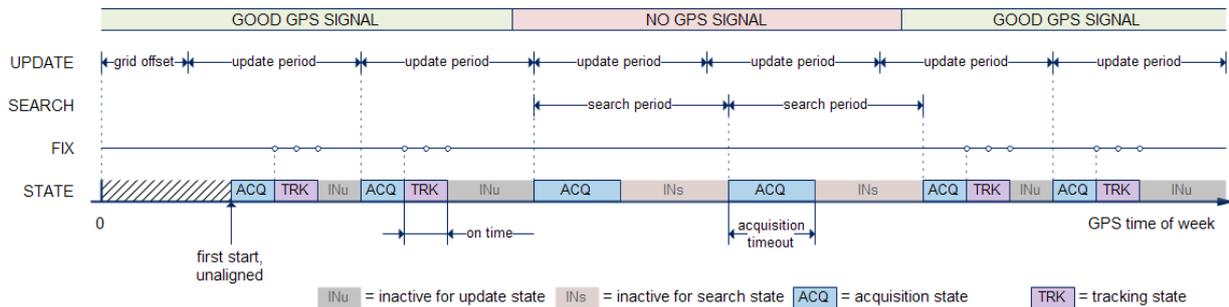
The following figure illustrates the state machine:

State machine

10.2.1.1 ON/OFF operation - long update period

When the receiver is switched on, it first enters *Acquisition* state. If it is able to obtain a valid 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 after the configured search period (minus a start-up margin). As soon as the receiver gets a valid position fix (one passing the [navigation output filters](#)), 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 (see section [Grid offset](#) for an explanation). 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

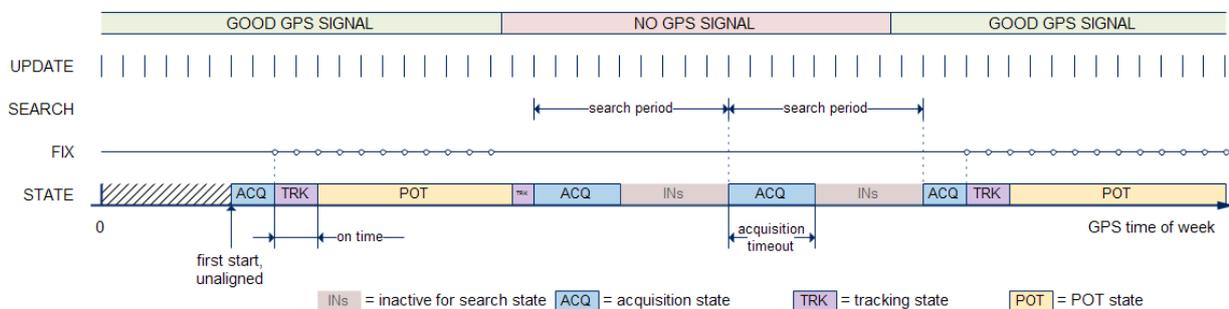


10.2.1.2 Cyclic tracking operation - short update period

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 valid position fix. 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



10.2.1.3 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 section [Wake-up](#) for more information on wake-up events.



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

10.2.1.4 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.

10.2.2 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 <i>Inactive for search</i> state
On-time	Time the receiver remains in <i>Tracking</i> state and produces position fixes
Wait for timefix	Wait for time fix before entering <i>Tracking</i> state
Do not enter <i>Inactive for search</i> state	Receiver does not enter <i>Inactive for search</i> 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

10.2.2.1 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. For long update periods (in the range of minutes or longer), only use ON/OFF operation.

See section [ON/OFF operation - long update period](#) and [Cyclic tracking operation - short update period](#) for more information on the two modes of operation.

10.2.2.2 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.



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.

10.2.2.3 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. If the GPS signal is very weak or absent, 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.

10.2.2.4 On time and wait for timefix

The *on-time* parameter specifies how long the receiver stays in *Tracking* state before switching to the *POT* state or *Inactive for update* state. 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 GPS 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.

10.2.2.5 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.

10.2.2.6 Update RTC and Ephemeris

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 Ephemeris option. The RTC is calibrated every 5 minutes and the ephemeris data is updated approximately every 30 minutes. See section [Satellite data download](#) for more information.

10.2.2.7 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) or *Acquisition/Tracking/POT* states (cyclic tracking operation). 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 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.

10.2.2.8 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 section [Use grid offset](#).



The grid offset is not used in cyclic tracking operation.

10.2.3 Features

10.2.3.1 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 up the receiver if 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. If the interval is too long, the receiver may return to *Inactive* state before the configuration message was received. It is therefore important to check for a [UBX-ACK-ACK](#) reply from the receiver to confirm that the configuration message was received.
- Send the configuration save message immediately after the configuration message.

10.2.3.2 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.

10.2.3.3 Behavior while USB host connected

As long as the receiver is connected to a USB host, it will not enter the lowest possible power state. This is because it must retain a small level of CPU activity to avoid breaching requirements of the USB specification. The drawback, however, is that power consumption is higher.



Wake-up by pin/UART 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.

10.2.3.4 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.

10.2.4 Examples

10.2.4.1 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 24×3600 s and the search period to 2×3600 s. 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 12×3600 s 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.

10.2.4.2 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.

10.3 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 (Continuous or Power Save Mode).

10.4 Power On/Off command

With message [RXM-PMREQ](#) the receiver can be forced to enter *Inactive* state (in Continuous and Power Save Mode). It will stay in *Inactive* 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.

10.5 EXTINT pin control when Power Save Mode is not active

The receiver can be forced OFF also when Power Save Mode is not active. This works the same way as [EXTINT pin control in Power Save Mode](#). Just as in Power Save Mode, this feature has to be enabled and configured using [CFG-PM2](#).

10.6 Measurement and navigation rate with Power Save Mode

In Continuous Mode, measurement and navigation rate is configured using [UBX-CFG-RATE](#). In Power Save Mode however, measurement and navigation rate can differ from the configured rates as follows:

- **Cyclic Operation:** When in state *Power Optimized Tracking*, the measurement and navigation rate is determined by the *updatePeriod* configured in [CFG-PM2](#). The receiver can however switch to *Tracking* state (e.g. to download data). When in *Tracking* state, the measurement and navigation rate is as configured with [UBX-CFG-RATE](#). Note: When the receiver is no longer able to produce position fixes, it can switch from Cyclic Operation to ON/OFF Operation (if this is not disabled with the *doNotEnterOff* switch in [CFG-PM2](#)). In that case the remarks below are relevant.
- **ON/OFF Operation:** When in state *Acquisition*, the measurement and navigation rate is **fixed to 2Hz**. All NMEA (an possibly UBX) messages that are output upon a navigation fix are also output with a rate of 2Hz.

This must be considered when choosing the baud rate of a receiver that uses Power Save Mode! Note that a receiver might stay in *Acquisition* state for quite some time (can be tens of seconds under weak signal conditions). When the receiver eventually switches to *Tracking* state, the measurement and navigation rate will be as configured with `UBX-CFG-RATE`.



When using Power Save Mode, the baud rate of the receiver must be chosen such that it can handle the amount of data that is output when measurement and navigation rate is 2Hz.

11 Time pulse



There is only limited support for the generation of time pulses when running in BeiDou mode. In particular the accuracy of the time pulse in BeiDou mode has not been calibrated.

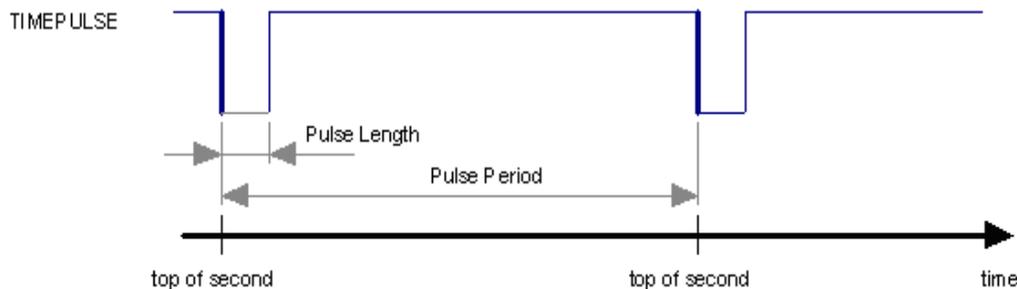
11.1 Introduction

u-blox GNSS receivers include a time pulse function providing clock pulses with configurable duration and frequency. The time pulse function can be configured using the `CFG-TP5` message. The `TIM-TP` message provides time information for the next pulse, time source and the quantization error of the output pin.

Pulse Mode: Rising



Pulse Mode: Falling



11.2 Recommendations

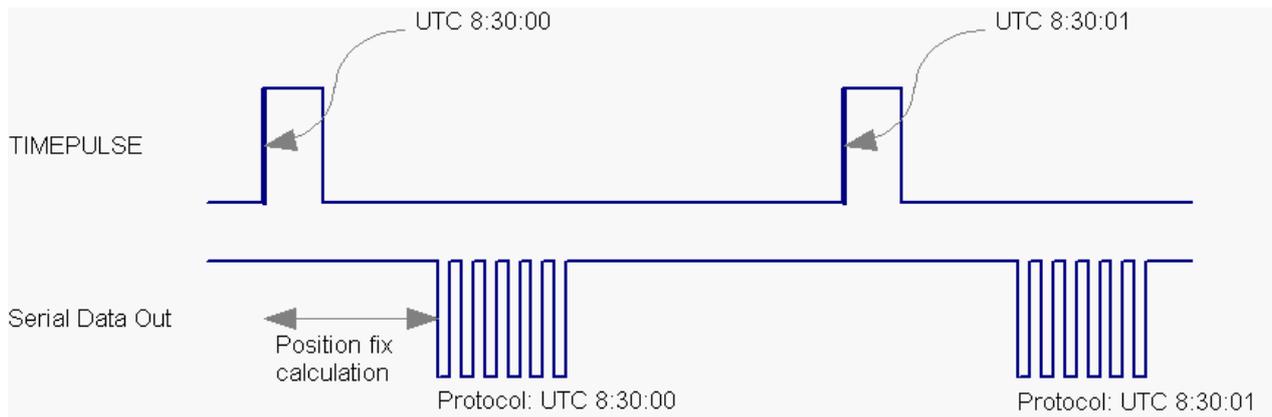
- For best time pulse performance it is recommended to disable the SBAS subsystem.
- Mixing different GNSS systems may impair the timing performance, thus for getting a consistent timing performance, all receivers used for synchronization should be configured to one and the same single GNSS, e.g. GPS only
- When using time pulse 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.
- 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 accordingly, by setting cable delay and user delay.
- The recommended configuration when using the `TIM-TP` message is to set both the measurement rate (`CFG-RATE`) and the time pulse frequency (`CFG-TP5`) to 1Hz.



Since the rate of `TIM-TP` is bound to the measurement rate, more than one `TIM-TP` message can appear between two pulses if the measurement rate is set larger than the time pulse frequency. In

this case all **TIM-TP** messages in between a time pulse $T1$ and $T2$ belong to $T2$ and the last **TIM-TP** before $T2$ reports the most accurate quantization error. In general, if the navigation solution rate and time pulse rate are configured to different values, there will not be a single **TIM-TP** message for each time pulse.

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.



11.3 Time pulse configuration

u-blox GNSS receivers provide one or two TIMEPULSE pins (dependant on product variant) delivering a time pulse (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 UTC time. Time pulse signals can be configured using the UBX proprietary message [CFG-TP5](#).

11.4 Configuring time pulse with UBX-CFG-TP5

The UBX message [CFG-TP5](#) can be used to change the time pulse settings, and includes the following parameters defining the pulse:

- **time pulse index** - Index of time pulse.
- **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 corresponding 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 corresponding flag is set to use another setting in locked mode.
- **user delay** - The cable delay from the receiver to the user device plus signal delay of any user application.
- **active** - time pulse 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 time pulse 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.
- **polarity** - If set, the first edge of the pulse is a rising edge (Pulse Mode: Rising).
- **grid UTC/GPS** - Selection between UTC (0) or GPS (1) timegrid. Also effects the time output by [TIM-TP](#) message.



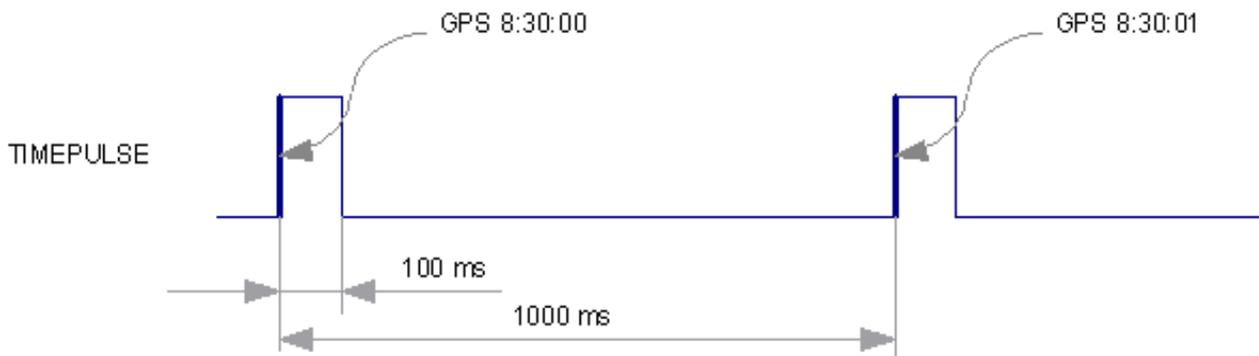
The maximum pulse length can't exceed the pulse period.



time pulse 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.

11.4.1 Example 1:

The example below shows the 1PPS TP signal generated on the time pulse 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.



UBX - CFG (Config) - TP5 (Timepulse 5)

Timepulse Settings

0 - TIMEPULSE

Active

Frequency Period

Period [us]

Length Duty Cycle

Length [us]

Lock to GPS Frequency if available

Other Setting in GPS time locked mode

Period Locked [us]

Length Locked [us]

Align Pulse to TO_W=0 as soon as GPS time is locked and valid

0 - UTC Time

Invert pulse polarity

User Delay [ns]

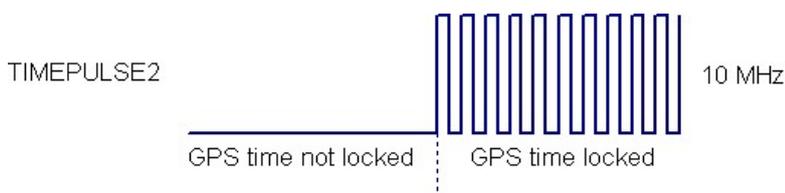
Receiver Global Settings

Cable Delay [ns]

RF Group Delay [ns]

11.4.2 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.



UBX - CFG (Config) - TP5 (Timepulse 5)

Timepulse Settings

1 - TIMEPULSE2

Active

Frequency Period

Frequency [Hz]

Length Duty Cycle

Duty [%]

Lock to GPS Frequency if available

Other Setting in GPS time locked mode

Frequency Locked [Hz]

Duty Locked [%]

Align Pulse to TOW=0 as soon as GPS time is locked and valid

0 - UTC Time

Invert pulse polarity

User Delay [ns]

Receiver Global Settings

Cable Delay [ns]

RF Group Delay [ns]

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 GNSS-specific.

The main purposes are

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

as well as error conditions.



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

Inbound data to the GNSS receiver is placed in buffers. Usage of these buffers is shown with the message [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.

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

Port Number assignment

Port #	Electrical Interface
0	DDC (I ² C compatible)
1	UART 1
3	USB
4	SPI

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

Protocol Number assignment

Protocol #	Protocol Name
0	UBX Protocol
1	NMEA Protocol

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 an unjammed environment to determine an appropriate value for the unjammed case. If the value rises significantly above this threshold, this indicates that a continuous wave jammer is present.

This indicator is always enabled.

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

12.3 Jamming/Interference Monitor (ITFM)

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
-------	----------------	-------------

Jamming/Interference monitor reported states continued

Value	Reported state	Description
0	Unknown	Jamming/interference 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 and interference is visible (above the thresholds); interference is probable reason why there is no fix

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



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



Jamming/Interference monitor is not supported in Power Save Mode (PSM) ON/OFF mode.

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

13 Timemark

The receiver can be used to provide an accurate measurement of the time at which a pulse was detected on the external interrupt pin. The reference time can be chosen by setting the time source parameter to GPS, UTC or local time in the `UBX-CFG-TP5` configuration message (using flags `LockGpsFreq` and `gridUtcGps`). The delay figures defined with `UBX-CFG-TP5` are also applied to the results output in the `UBX-TIM-TM2` message.

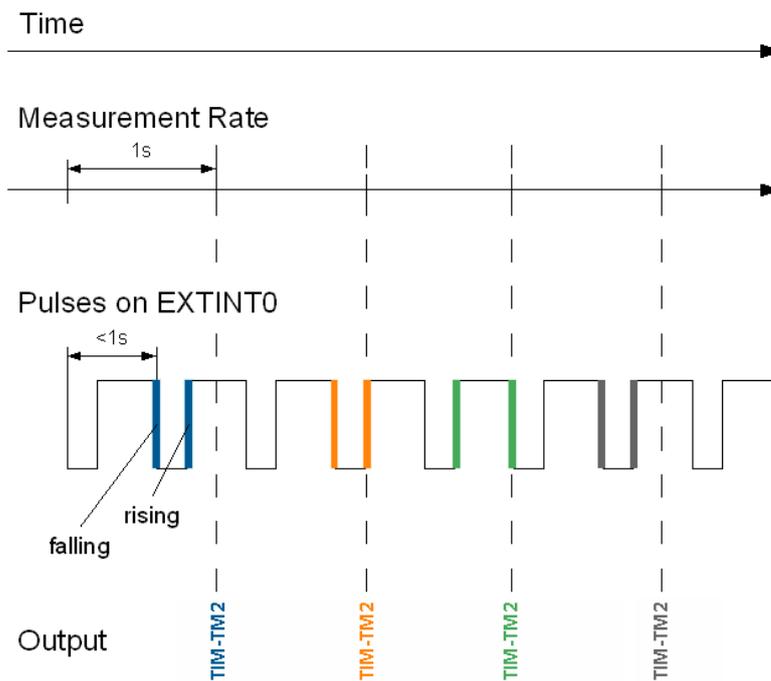
A `UBX-TIM-TM2` message is output at the next epoch if

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

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



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



14 Multi-GNSS Assistance (MGA)

14.1 Introduction

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

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

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

14.2 Assistance Data

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

- **Position:** Estimated receiver position can be submitted to the receiver using the [UBX-MGA-INI-POS_XYZ](#) or [UBX-MGA-INI-POS_LLH](#) messages.
- **Time:** The current time can either be supplied as an inexact value via the standard communication interfaces, suffering from latency depending on the baud rate, or using hardware time synchronization where an

accurate time pulse is connected to an external interrupt. The preferred option is to supply UTC time using the [UBX-MGA-INI-TIME_UTC](#) message, but times referenced to some GNSS can be delivered with the [UBX-MGA-INI-TIME_GNSS](#) message.

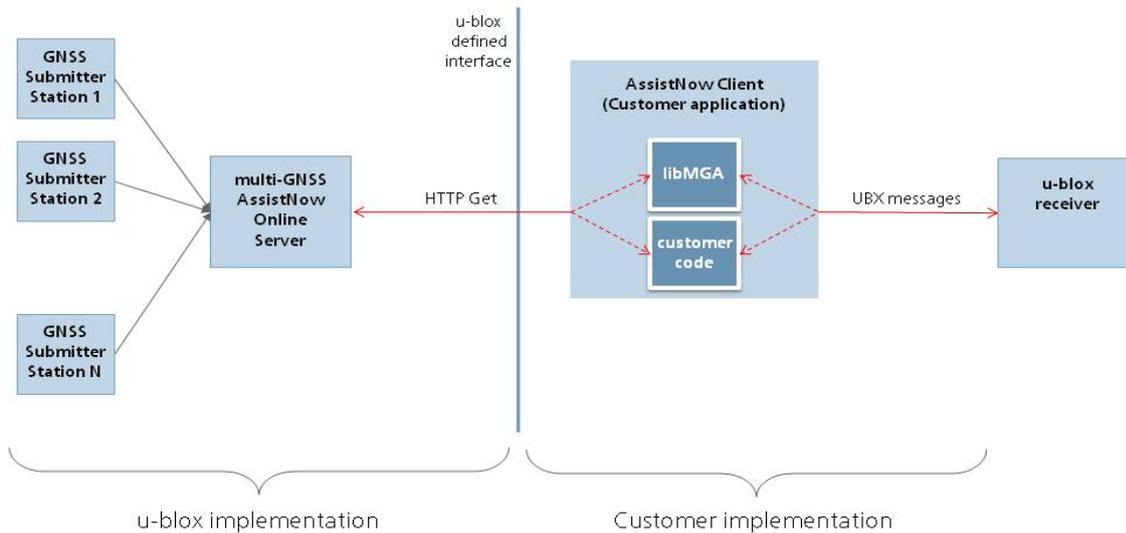
- **Clock drift:** An estimate of the clock drift can be sent to the receiver using the [UBX-MGA-INI-CLKD](#) message.
- **Frequency:** It is possible to supply hardware frequency aiding by connecting a periodic rectangular signal with a frequency up to 500 kHz and arbitrary duty cycle (low/high phase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

14.3 AssistNow Online

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

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

multi-GNSS AssistNow Online Architecture



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



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

14.3.1 Host Software

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

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

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



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

14.3.2 AssistNow Online Sequence

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

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

14.3.3 Flow Control

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

Flow control is activated by setting the `ackAiding` parameter in the `UBX-CFG-NAVX5` message. As a result the receiver will issue an acknowledgement message (`UBX-MGA-ACK`) for each assistance message it successfully receives. The host software can examine these acknowledgements to establish whether there were any problems with the data sent to the receiver and deduce (by the lack of acknowledgement) if any messages have been lost. It may then be appropriate to resend some of the assistance messages.

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

14.3.4 Authorization

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

14.3.5 Service Parameters

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

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

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

The following rules apply:

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

The following table describes the keys that are supported.

AssistNow Online Parameter Keys

Key Name	Unit/Range	Optional	Description
token	String	Mandatory	The authorization token supplied by u-blox when a client registers to use the service.
gnss	String	Mandatory	A comma separated list of the GNSS for which data should be returned. Valid GNSS are: gps, qzss and glo.

AssistNow Online Parameter Keys continued

Key Name	Unit/Range	Optional	Description
datatype	String	Mandatory	A comma separated list of the data types required by the client. Valid data types are: eph, alm, aux and pos. Time data is always returned for each request. If the value of this parameter is an empty string, only time data will be returned.
lat, lon	Numeric [degrees]	Optional	Approximate user position in WGS-84 Latitude / Longitude in units of degrees and fractional degrees. Example: lat=47.2;lon=8.55.
alt	Numeric [meters]	Optional	Approximate user altitude above WGS84 Ellipsoid in units of meters. If this value is not provided, the server assumes an altitude of 0 meters.
pacc	Numeric [meters]	Optional	Approximate accuracy of submitted position (see position parameters note below). If this value is not provided, the server assumes an accuracy of 300km.
tacc	Numeric [seconds]	Optional	The timing accuracy (see time parameters note below). If this value is not provided, the server assumes an accuracy of 10 seconds.
latency	Numeric [seconds]	Optional	Typical latency between the time the server receives the request, and the time when the assistance data arrives at the GNSS receiver. The server can use this value to correct the time being transmitted to the client. If this value is not provided, the server assumes a latency of 0.
filteronpos	(no value required)	Optional	If present, the ephemeris data returned to the client will only contain data for the satellites which are likely to be visible from the approximate position provided by the lat, lon, alt and pacc parameters. If the lat/lon parameters are not provided the service will return an error.
filteronsv	String	Optional	A comma separated list of u-blox gnssId:svId pairs. The ephemeris data returned to the client will only contain data for the listed satellites.

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

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

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

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

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

14.3.5.2 Time parameters (tacc and latency)

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

network latency and any time between the client receiving the data and it being provided to the receiver.

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

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

14.3.6 Multiple Servers

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

14.4 AssistNow Offline

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



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

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

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

The data obtained from the AssistNow Offline Service is organised by date, normally a day at a time.

Consequently the more weeks for which coverage is requested, the larger the amount of data to handle.

Similarly, each different GNSS requires its own data and in the extreme cases, several hundred kilobytes of data will be provided by the service. This amount can be reduced by requesting lower resolution, but this will have a small negative impact on both position accuracy and TTFF. See the section on [Offline Service Parameters](#) for details of how to specify these options.

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

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

14.4.1 Service Parameters

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

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

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

The following rules apply:

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

The following table describes the keys that are supported.

AssistNow Offline Parameter Keys

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

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

token=XXXXXXXXXXXXXXXXXXXXXXXXXX;gnss=gps,glo;

14.4.2 Authorization

The AssistNow Offline Service uses the same authorization process as AssistNow Online; see [above](#) for details.

14.4.3 Multiple Servers

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

14.4.4 Time, Position and Almanac

While AssistNow Offline can be used on its own, it is expected that the user will provide estimates of the receiver's current position, the current time and ensure that a reasonably up to date almanac is available. In most cases this information is likely to be available without the user needing to do anything. For example,

where the receiver is connected to a battery backup power supply and has a functioning real time clock (RTC), the receiver will keep its own sense of time and will retain the last known position and any almanac. However, should the receiver be completely unpowered before startup, then it will greatly improve TTFF if time, position and almanac can be supplied in some form.

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

Obviously, where a receiver has a functioning RTC, it should be able to keep its own sense of time, but where no RTC is fitted (or power is completely turned off), providing a time estimate via the [UBX-MGA-INI-TIME_UTC](#) message will be beneficial.

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

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

14.4.5 Flash-based AssistNow Offline

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

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



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

14.4.5.1 Flash-based Storage Procedure

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

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

Note that the final block may be smaller than 512 bytes (where the total data size is not perfectly divisible by

512). Also, the [UBX-MGA-FLASH-ACK](#) messages are distinct from the [UBX-MGA-ACK](#) messages used for other AssistNow functions.

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

14.4.6 Host-based AssistNow Offline

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

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

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

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

14.4.6.1 Host-based Procedure

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

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

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

14.5 Preserving Information During Power-off

The performance of u-blox receivers immediately after they are turned on is enhanced by providing them with as much useful information as possible. Assistance (both [Online](#) and [Offline](#)) is one way to achieve this, but retaining information from previous use of the receiver can be just as valuable. All the [types of data delivered](#)

by assistance can be retained while the receiver is powered down for use when power is restored. Obviously the value of this data will diminish as time passes, but in many cases it remains very useful and can significantly improve time to first fix.

There are several ways in which a u-blox receiver can retain useful data while it is powered down, including:

- **Battery Backed RAM:** The receiver can be supplied with sufficient power to maintain a small portion of internal storage, while it is otherwise turned off. This is the best mechanism, provided that the small amount of electrical power required can be supplied continuously.
- **Save on Shutdown:** The receiver can be instructed to dump its current state to the attached flash memory (where fitted) as part of the shutdown procedure; this data is then automatically retrieved when the receiver is restarted. See the description of the [UBX-UPD-SOS](#) messages for more information.
- **Database Dump:** The receiver can be asked to dump the state of its internal database in the form of a sequence of UBX messages reported to the host; these messages can be stored by the host and then sent back to the receiver when it has been restarted. See the description of the [UBX-MGA-DBD](#) messages for more information.

14.6 AssistNow Autonomous

14.6.1 Introduction

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

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



The AssistNow Autonomous feature is disabled by default. It can be enabled using the [UBX-CFG-NAVX5](#) message.

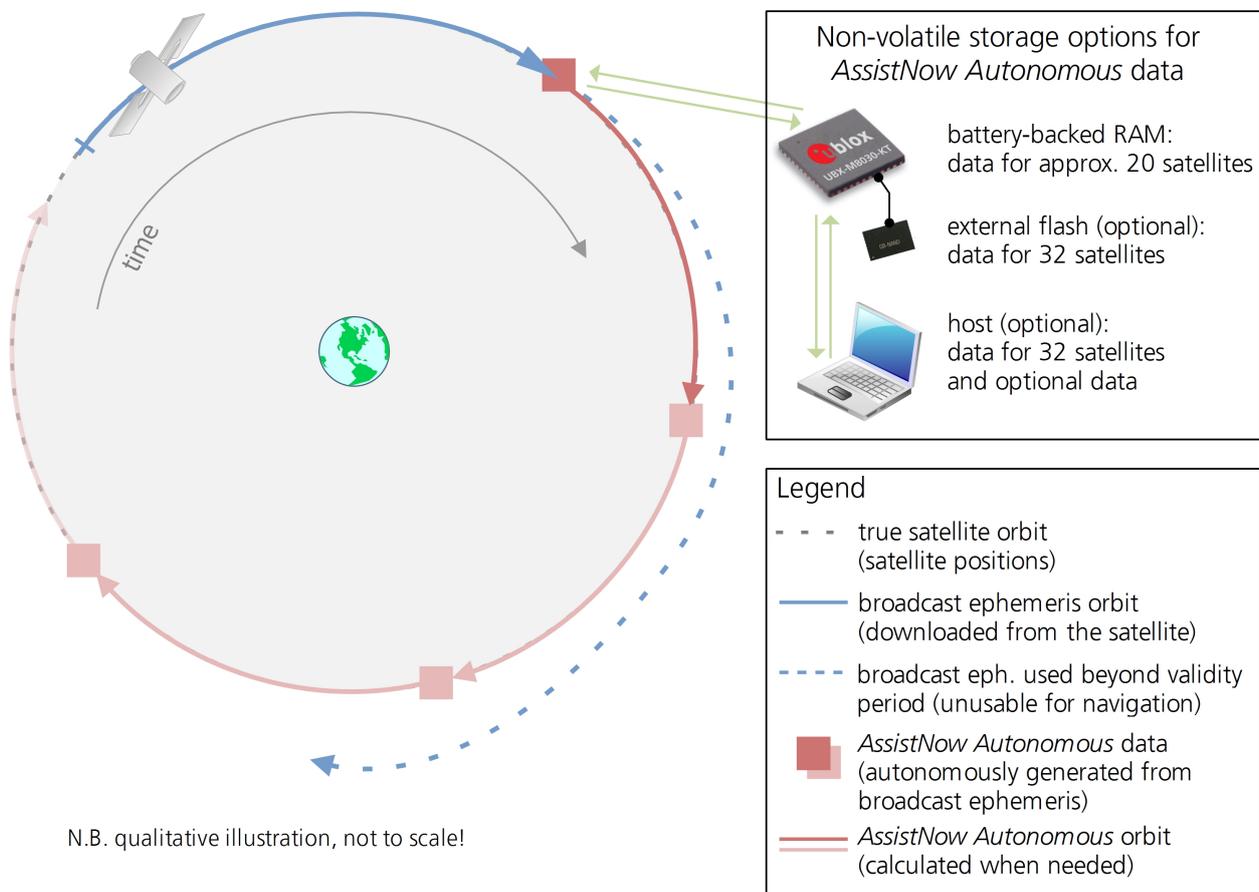
14.6.2 Concept

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

- A broadcast ephemeris downloaded from the satellite is a precise representation of a part (for GPS nominally four hours) of the satellite's true orbit (trajectory). It is not usable for positioning beyond this validity period because it diverges dramatically from the true orbit afterwards.
- The *AssistNow Autonomous orbit* is an extension of one or more broadcast ephemerides. It provides a long-term orbit for the satellite for several revolutions. Although this orbit is not perfectly precise it is a sufficiently accurate representation of the true orbit to be used for navigation.
- The *AssistNow Autonomous data* is automatically and autonomously generated from downloaded (or assisted) ephemerides. The data is stored automatically in the on-chip battery-backed memory (BBR). Optionally, the data can be backed-up in external flash memory or on the host. The number of satellites for which data can be stored depends on the receiver configuration and may change during operation.
- If no broadcast ephemeris is available for navigation *AssistNow Autonomous* automatically generates the required parts of the orbits suitable for navigation from the stored data. The data is also automatically kept current in order to minimize the calculation time once the navigation engine needs orbits.
- The operation of the *AssistNow Autonomous* feature is transparent to the user and the operation of the

receiver. All calculations are done in background and do not affect the normal operation of the receiver.

- The *AssistNow Autonomous* subsystem automatically invalidates data that has become too old and that would introduce unacceptable positioning errors. This threshold is configurable (see below).
- The prediction quality will be automatically improved if the satellite has been observed multiple times. However, this requires the availability of a suitable flash memory (see the *Hardware Integration Manual* for a list of supported devices). Improved prediction quality also positively affects the maximum usability period of the data.
- *AssistNow Autonomous* considers GPS and GLONASS satellites only. For GLONASS support a suitable flash memory is mandatory because a single broadcast ephemeris spans to little of the orbit (only approx. 30 minutes) in order to extend it in a usable way. Only multiple observations of the same GLONASS satellite that span at least four hours will be used to generate data.



14.6.3 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 (for GPS satellites observed once) and up to twelve days (for GPS and GLONASS satellites observed multiple times over a period of at least half a day).

- Note that disabling the *AssistNow Autonomous* feature will delete all previously collected satellite observation data from the flash memory.
- The [UBX-NAV-AOPSTATUS](#) message provides information on the current state of the *AssistNow Autonomous* subsystem. The status indicates whether the *AssistNow Autonomous* subsystem is currently idle (or not enabled) or busy generating data or orbits. Hosts should monitor this information and only power-off the receiver when the subsystem is idle (that is, when the `status` field shows a steady zero).
- The [UBX-NAV-SAT](#) message indicates the use of *AssistNow Autonomous* orbits for individual satellites.
- The [UBX-NAV-ORB](#) message indicates the availability of *AssistNow Autonomous* orbits for individual satellites.
- The [UBX-MGA-DBD](#) message provides a means to retrieve the *AssistNow Autonomous* data from the receiver in order to preserve the data in power-off mode where no battery backup is available. Note that the receiver requires the absolute time (i.e. full date and time) to calculate *AssistNow Autonomous* orbits. For best performance it is, therefore, recommended to supply this information to the receiver using the [UBX-MGA-INI-TIME.UTC](#) message in this scenario.

14.6.4 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 (see the *Interface* section above), and that it knows which satellites are visible, either from the almanac or from tracking the respective signals.

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

AssistNow Autonomous will typically extend a broadcast ephemeris for up to three to twelve days. The [UBX-CFG-NAVX5](#) (see above) message allows changing this threshold by setting the «maximum acceptable modelled orbit error» (in meters). Note that this number does not reflect the true orbit error introduced by extending the ephemeris. It is a statistical value that represents a certain expected upper limit based on a number of parameters. A rough approximation that relates the maximum extension time to this setting is: $maxError [m] = maxAge [d] * f$, where the factor f is 30 for data derived from satellites seen once and 10 for data derived for satellites seen multiple time during a long enough time period (see the *Concept* section above).

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

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

The repeatability of the satellite constellation is a potential pitfall for the use of the *AssistNow Autonomous* feature. For a given location on Earth the (GPS) constellation (geometry of visible satellites) repeats every 24 hours. Hence, when the receiver «learned» about a number of satellites at some point in time the same

satellites will in most places *not* be visible 12 hours later, and the available *AssistNow Autonomous* data will not be of any help. Again 12 hours later, however, usable data would be available because it had been generated 24 hours ago.

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

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



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

15 Odometer

15.1 Introduction

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



The odometer feature is disabled by default. It can be enabled using the [UBX-CFG-ODO](#) message.

15.2 Odometer Output

The odometer output is published in the [UBX-NAV-ODO](#) message. This message contains the following elements:

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

If logging is enabled, then the odometer ground distance value will be included in logged position data.

15.3 Odometer Configuration

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

- *Running*: the algorithm is optimized for typical dynamics encountered while running, i.e the Doppler-based velocity solution is assumed to be of lower quality;

- *Cycling*: the algorithm is optimized for typical dynamics encountered while cycling;
- *Swimming*: the algorithm is optimized for very slow and smooth trajectories typically encountered while swimming;
- *Car*: the algorithm assumes that good Doppler measurements are available (i.e. the antenna is subject to low vibrations) and is optimized for typical dynamics encountered by cars.



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

15.4 Resetting the Odometer

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

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

16 Logging

16.1 Introduction

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

The following tables list all the logging related messages:

Logging control and configuration messages

Message	Description
UBX-LOG-CREATE	Creates a log file and activates the logging subsystem
UBX-LOG-ERASE	Erases a log file and deactivates the logging subsystem
UBX-CFG-LOGFILTER	Used to start/stop recording and set/get the logging configuration
UBX-LOG-INFO	Provides information about the logging system
UBX-LOG-STRING	Enables a host process to write a string of bytes to the log file

Logging retrieval messages

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

16.2 Setting the logging system up

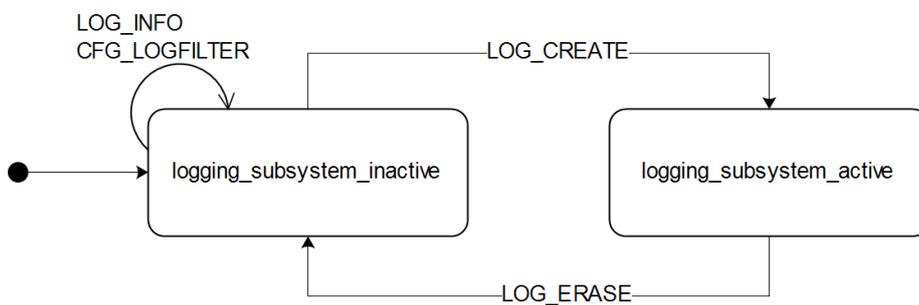
An empty log can be created using the [UBX-LOG-CREATE](#) message and a log can be deleted with the [UBX-LOG-ERASE](#) message. The logging system will only be running if a log is in existence, so most logging messages will be rejected with an [UBX-ACK-NAK](#) message if there is no log present. Only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

somewhat smaller than the size specified.

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

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

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



16.3 Information about the log

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

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

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

Log overhead size

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

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

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

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

16.4 Recording

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

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

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

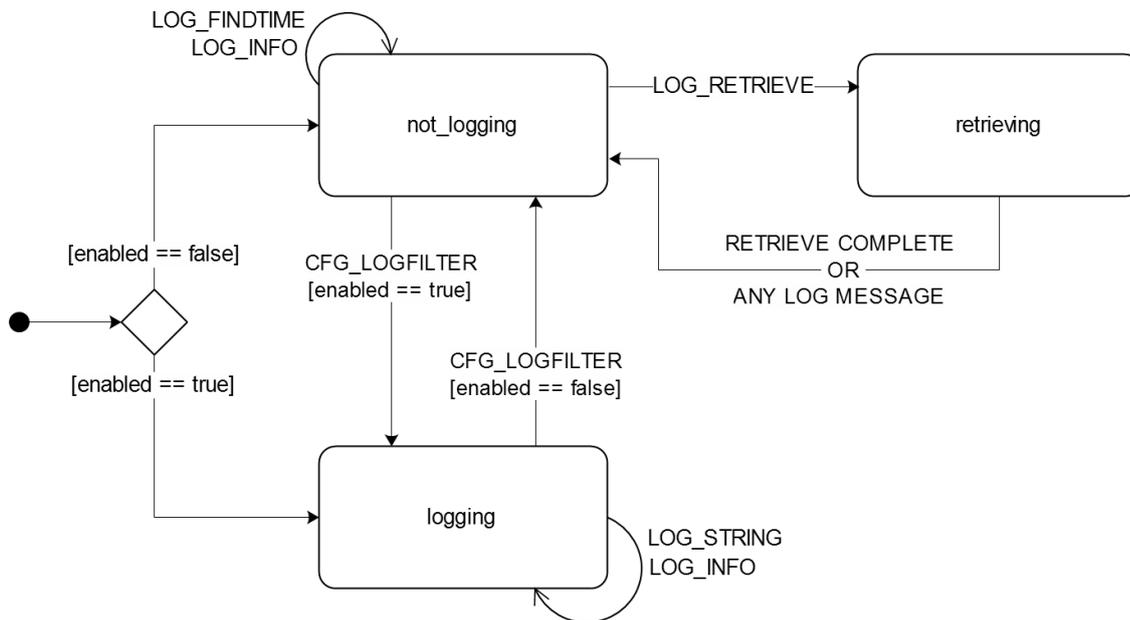
Position fixes are compressed to economise on the amount of flash space used. In order to improve the compression, the fix values are rounded to improve their compression. This means that the values returned by the logging system may differ slightly from any which are gathered in real time.

In `On/Off Power Save Mode` it is possible to configure the logging system so that only one fix is recorded for each on period. This will be recorded immediately before the receiver powers off and will be the best fix seen during the on period (in this case, "best" is defined as being the fix with the lowest horizontal accuracy figure).

The recorded data for a fix comprises :

- The time and date of the fix recorded to a precision of one second
- Latitude and longitude to a precision of one millionth of a degree. Depending on position on Earth this is a precision in the order of 0.1m
- Altitude (height above mean sea level) to a precision of 1m
- Ground speed to a precision of 1 cm/s
- The fix type (only successful fix types, since these are the only ones recorded)
- The number of satellites used in the fix is recorded, but no value greater than 19 is logged; a value of 19 means 19 or more satellites
- A horizontal accuracy estimate is recorded to give an indication of fix quality
- Heading to a precision of one degree
- Odometer distance data (if odometer is enabled)

The states of the active logging subsystem



16.5 Retrieval

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

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

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

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

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

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

16.6 Command message acknowledgement

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

It is possible to send a small number of logging commands without waiting for acknowledgement, since there

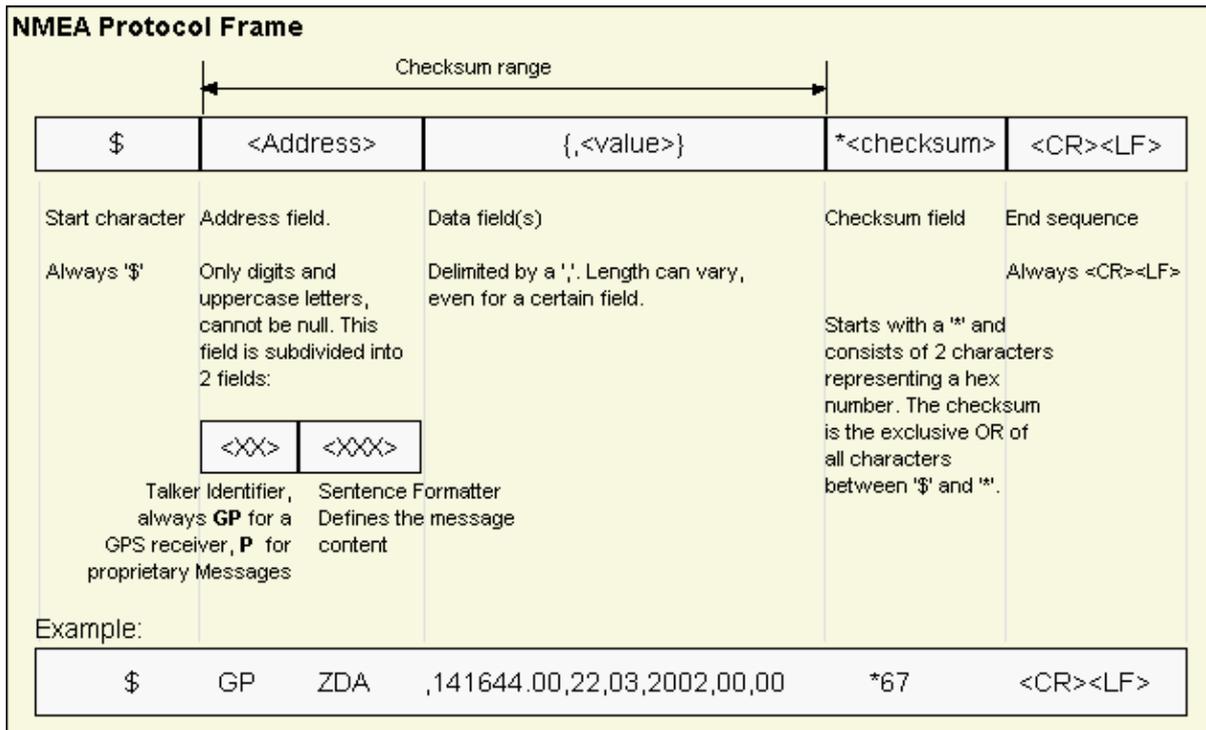
is a command queue, but this risks confusion between the acknowledgements for the commands. Also a command queue overflow would result in commands being lost.

NMEA Protocol

17 Protocol Overview

17.1 Message Format

NMEA messages sent by the GNSS receiver are based on NMEA 0183 Version 4.0. The following picture shows the structure of a NMEA protocol message.



For further information on the NMEA Standard, refer to *NMEA 0183 Standard For Interfacing Marine Electronic Devices*, Version 4.00, November 1, 2008. 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.

17.2 Talker ID

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

NMEA Talker IDs

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

17.3 Protocol Configuration

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

There are four NMEA standards supported. The default NMEA version is 4.0. Alternatively versions 4.1, 2.3, and 2.1 can be enabled (for details on how this affects the output refer to section [Position Fix Flags in NMEA Mode](#)).

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

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

NMEA filtering flags

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

NMEA flags

<i>Parameter</i>	<i>Description</i>
Compatibility Mode	Some older NMEA applications expect the NMEA output to be formatted in a specific way, for example, they will only work if the latitude and longitude have exactly four digits behind the decimal point. u-blox receivers offer a compatibility mode to support these legacy applications.
Consideration Mode	u-blox receivers use a sophisticated signal quality detection scheme, in order to produce the best possible position output. This algorithm considers all SV measurements, and may eventually decide to only use a subset thereof, if it improves the overall position accuracy. If Consideration mode is enabled, all satellites, which were considered for navigation, are communicated as being used for the position determination. If Consideration Mode is disabled, only those satellites which after the consideration step remained in the position output are marked as being used.
Limit82 Mode	Enabling this mode will limit the NMEA sentence length to a maximum of 82 characters.

Extended configuration

<i>Option</i>	<i>Description</i>
GNSS to filter	Filters satellites based on their GNSS
Satellite numbering	This field configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID.

Extended configuration continued

Option	Description
Main Talker ID	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see UBX-CFG-GNSS). This field enables the main Talker ID to be overridden.
GSV Talker ID	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden.
BDS Talker ID	By default the Talker ID for BeiDou is 'GB'. This field enables the BeiDou Talker ID to be overridden.

17.4 Satellite Numbering

The NMEA protocol (V4.0) identifies satellites with a two digit number, reserving the numbers 1 to 32 for GPS, 33-64 for SBAS and 65-96 for GLONASS. So, for example, GLONASS SV4 is reported using number 68. u-blox receivers support this method in their NMEA output when "strict" SV numbering is selected. In most cases this is the default setting, but can be checked or set using [UBX-CFG-NMEA](#).

Unfortunately there is currently no standard way of identifying satellites from any other GNSS within the NMEA protocol. In order to support QZSS within current receivers and prepare for support of other systems (e.g. Galileo) in future receivers, an "extended" SV numbering scheme can be enabled (using [UBX-CFG-NMEA](#)). This uses the NMEA-defined numbers where possible, but adds other number ranges to support other GNSS. Note however that these non-standard extensions require 3 digit numbers, which may not be supported by some NMEA parsing software. For example QZSS satellites are reported using numbers in the range 193 to 197.

See [Satellite Numbering Summary](#) for a complete list of satellite numbers.



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

17.5 Latitude and Longitude Format

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

Latitude 47 Degrees, 17.112671 Minutes

Longitude 8 Degrees, 33.914843 Minutes

or

Latitude 47 Degrees, 17 Minutes, 6.76026 Seconds

Longitude 8 Degrees, 33 Minutes, 54.89058 Seconds

or

Latitude 47.28521118 Degrees

Longitude 8.56524738 Degrees

17.6 Position Fix Flags

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

Flags in NMEA 2.3 and above

Flags in NMEA 2.3 and above continued

NMEA Message: Field	No position fix (at power-up, after losing satellite lock)	GNSS fix, but user limits exceeded	Dead reckoning fix, but user limits exceeded	Dead reckoning fix (ADR with external sensors, linear extrapolation, or map matching)	2D GNSS fix	3D GNSS fix	Combined GNSS/dead reckoning fix (ADR with external sensors)
NMEA Message: Field	No position fix (at power-up, after losing satellite lock)	GNSS fix, but user limits exceeded	Dead reckoning fix, but user limits exceeded	Dead reckoning fix (ADR with external sensors, linear extrapolation, or map matching)	2D GNSS fix	3D GNSS fix	Combined GNSS/dead reckoning fix (ADR with external sensors)
GLL, RMC: status	V	V	V	A	A	A	A
	V=Data Invalid, A=Data Valid						
GGA: quality	0	0	6	6	1 / 2	1 / 2	1 / 2
	0=No Fix, 1=Autonomous GNSS Fix, 2=Differential GNSS Fix, 6=Estimated/Dead Reckoning Fix						
GSA: navMode	1	1	2	2	2	3	3
	1=No Fix, 2=2D Fix, 3=3D Fix						
GLL, RMC, VTG, GNS: posMode	N	N	E	E	A / D	A / D	A / D
	N=No Fix, E=Estimated/Dead Reckoning Fix, A=Autonomous GNSS Fix, D=Differential GNSS Fix						

Flags in NMEA 2.1 and below

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

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

Extra fields in NMEA 4.1 and above

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

17.7 Multi-GNSS considerations

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

NMEA output for Multi-GNSS

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

NMEA output for Multi-GNSS continued

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

17.8 Output of Invalid/Unknown Data

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

A valid position fix is reported as follows:

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

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

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

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

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

Note:



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



Output of invalid data marked with the 'Invalid/Valid' Flags can be enabled 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).

17.9 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 Standard Messages		Standard Messages	
59	DTM	0xF0 0x0A	Datum Reference
60	GBQ	0xF0 0x44	Poll a standard message (if the current Talker ID is GB)
60	GBS	0xF0 0x09	GNSS Satellite Fault Detection
61	GGA	0xF0 0x00	Global positioning system fix data
62	GLL	0xF0 0x01	Latitude and longitude, with time of position fix and status
63	GLQ	0xF0 0x43	Poll a standard message (if the current Talker ID is GL)
64	GNQ	0xF0 0x42	Poll a standard message (if the current Talker ID is GN)
64	GNS	0xF0 0x0D	GNSS fix data
65	GPQ	0xF0 0x40	Poll a standard message (if the current Talker ID is GP)
66	GRS	0xF0 0x06	GNSS Range Residuals
67	GSA	0xF0 0x02	GNSS DOP and Active Satellites
68	GST	0xF0 0x07	GNSS Pseudo Range Error Statistics
69	GSV	0xF0 0x03	GNSS Satellites in View
70	RMC	0xF0 0x04	Recommended Minimum data
71	TXT	0xF0 0x41	Text Transmission

NMEA Messages Overview continued

Page	Mnemonic	Cls/ID	Description
72	VLW	0xF0 0x0F	Dual ground/water distance
72	VTG	0xF0 0x05	Course over ground and Ground speed
73	ZDA	0xF0 0x08	Time and Date
NMEA PUBX Messages		Proprietary Messages	
75	CONFIG	0xF1 0x41	Set Protocols and Baudrate
76	POSITION	0xF1 0x00	Lat/Long Position Data
77	SVSTATUS	0xF1 0x03	Satellite Status
78	TIME	0xF1 0x04	Time of Day and Clock Information

18 Standard Messages

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

18.1 DTM

18.1.1 Datum Reference

<i>Message</i>	DTM		
<i>Description</i>	Datum Reference		
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
<i>Type</i>	Output Message		
<i>Comment</i>	This message gives the difference between the current datum and the reference datum. The current datum defaults to WGS84 The reference datum cannot be changed and is always set to WGS84.		
<i>Message Info</i>	<i>ID for CFG-MSG</i>	<i>Number of fields</i>	
	0xF0 0x0A	11	

Message Structure:

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

Example:

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

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

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

18.2 GBQ

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

Message	GBQ		
Description	Poll a standard message (if the current Talker ID is GB)		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Input Message		
Comment	Polls a standard NMEA message if the current Talker ID is GB		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x44	4	

Message Structure:

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

Example:

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

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

18.3 GBS

18.3.1 GNSS Satellite Fault Detection

Message	GBS		
Description	GNSS Satellite Fault Detection		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	This message outputs the results of the Receiver Autonomous Integrity Monitoring Algorithm (RAIM). <ul style="list-style-type: none"> The fields errLat, errLon and errAlt output the standard deviation of the position calculation, using all satellites which pass the RAIM test successfully. The fields errLat, errLon and errAlt are only output if the RAIM process passed successfully (i.e. no or successful edits happened). These fields are never output if 4 or fewer satellites are used for the navigation calculation (because, in such cases, integrity can not be determined by the receiver autonomously). The fields prob, bias and stddev 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	13	

Message Structure:

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

Example:

```
$GPGGBS,235503.00,1.6,1.4,3.2,,,,,*40
$GPGGBS,235458.00,1.4,1.3,3.1,03,, -21.4,3.8,1,0*5B
```

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

18.4 GGA

18.4.1 Global positioning system fix data

Message	GGA		
Description	Global positioning system fix data		
Firmware	Supported on: • u-blox M8 firmware version 2.00		
Type	Output Message		
Comment	<p>The output of this message is dependent on the currently selected datum (default: WGS84). The NMEA specification indicates that the GGA message is GPS specific. However, when the receiver is configured for multi-GNSS, the GGA message contents will be generated from the multi-GNSS solution. For multi-GNSS use, it is recommended that the NMEA-GNS message is used instead.</p> <p>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.).</p>		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x00	17	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
-----------	------	------	--------	---------	-------------

GGA continued

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

Table Quality Indicator

Quality Indicator	Description, see also position fix flags description
0	No Fix / Invalid
1	Standard GPS (2D/3D)
2	Differential GPS
6	Estimated (DR) Fix

18.5 GLL
18.5.1 Latitude and longitude, with time of position fix and status

Message	GLL		
Description	Latitude and longitude, with time of position fix and status		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
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	10	

Message Structure:

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

Example:

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

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

18.6 GLQ

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

Message	GLQ		
Description	Poll a standard message (if the current Talker ID is GL)		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Input Message		
Comment	Polls a standard NMEA message if the current Talker ID is GL		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x43	4	

Message Structure:

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

Example:

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

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

18.7 GNQ

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

Message	GNQ		
Description	Poll a standard message (if the current Talker ID is GN)		
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 		
Type	Input Message		
Comment	Polls a standard NMEA message if the current Talker ID is GN		
Message Info	<i>ID for CFG-MSG</i>	<i>Number of fields</i>	
	0xF0 0x42	4	

Message Structure:

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

Example:

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

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

18.8 GNS

18.8.1 GNSS fix data

Message	GNS		
Description	GNSS fix data		
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	The output of this message is dependent on the currently selected datum (default: WGS84) Time and position, together with GNSS fixing related data (number of satellites in use, and the resulting HDOP, age of differential data if in use, etc.).		
Message Info	<i>ID for CFG-MSG</i>	<i>Number of fields</i>	
	0xF0 0x0D	16	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxGNS	-	string	\$GPGNS	GNS Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	091547.00	UTC time, see note on UTC representation

GNS continued

Field No.	Name	Unit	Format	Example	Description
2	lat	-	ddmm. mmmm	5114.50897	Latitude (degrees & minutes), see format description
3	NS	-	character	N	North/South indicator
4	long	-	dddmm. mmmm	00012.28663	Longitude (degrees & minutes), see format description
5	EW	-	character	E	East/West indicator
6	posMode	-	character	AA	Positioning mode, see position fix flags description . First character for GPS, second character for GLONASS
7	numSV	-	numeric	10	Number of satellites used (range: 0-99)
8	HDOP	-	numeric	0.83	Horizontal Dilution of Precision
9	alt	m	numeric	111.1	Altitude above mean sea level
10	sep	m	numeric	45.6	Geoid separation: difference between geoid and mean sea level
11	diffAge	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
12	diffStation	-	numeric	-	ID of station providing differential corrections (blank when DGPS is not used)
13	navStatus	-	character	V	Navigational status indicator (V = Equipment is not providing navigational status information) NMEA v4.1 and above only
14	cs	-	hexadecimal	*71	Checksum
15	<CR><LF>	-	character	-	Carriage return and line feed

18.9 GPQ

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

Message	GPQ		
Description	Poll a standard message (if the current Talker ID is GP)		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Input Message		
Comment	Polls a standard NMEA message if the current Talker ID is GP		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x40	4	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxGPQ	-	string	\$EIGPQ	GPQ Message ID (xx = Talker ID of the device requesting the poll)
1	msgId	-	string	RMC	Message ID of the message to be polled

GPQ continued

Field No.	Name	Unit	Format	Example	Description
2	cs	-	hexadecimal	*3A	Checksum
3	<CR><LF>	-	character	-	Carriage return and line feed

18.10 GRS

18.10.1 GNSS Range Residuals

Message	GRS		
Description	GNSS Range Residuals		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	<p>This messages relates to associated GGA and GSA messages.</p> <p>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.</p> <p>In a multi-GNSS system this message will be output multiple times, once for each GNSS.</p>		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x06	19	

Message Structure:

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

Example:

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

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

18.11 GSA

18.11.1 GNSS DOP and Active Satellites

Message	GSA		
Description	GNSS DOP and Active Satellites		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	The GPS receiver operating mode, satellites used for navigation, and DOP values. <ul style="list-style-type: none"> If less than 12 SVs are used for navigation, the remaining fields are left empty. If more than 12 SVs are used for navigation, only the IDs of the first 12 are output. The SV numbers (fields 'sv') are in the range of 1 to 32 for GPS satellites, and 33 to 64 for SBAS satellites (33 = SBAS PRN 120, 34 = SBAS PRN 121, and so on) In a multi-GNSS system this message will be output multiple times, once for each GNSS.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x02	21	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxGSA	-	string	\$GPGSA	GSA Message ID (xx = current Talker ID)
1	opMode	-	character	A	Operation mode, see first table below
2	navMode	-	digit	3	Navigation mode, see second table below and position fix flags description
Start of repeated block (12 times)					
3 + 1*N	sv	-	numeric	29	Satellite number
End of repeated block					
15	PDOP	-	numeric	1.94	Position dilution of precision
16	HDOP	-	numeric	1.18	Horizontal dilution of precision
17	VDOP	-	numeric	1.54	Vertical dilution of precision
18	systemId	-	numeric	1	NMEA defined GNSS System ID NMEA v4.1 and above only
19	cs	-	hexadecimal	*0D	Checksum
20	<CR><LF>	-	character	-	Carriage return and line feed

Table Operation Mode

Operation Mode	Description
M	Manually set to operate in 2D or 3D mode
A	Automatically switching between 2D or 3D mode

Table Navigation Mode

Navigation Mode	Description, see also position fix flags description
1	Fix not available
2	2D Fix
3	3D Fix

18.12 GST

18.12.1 GNSS Pseudo Range Error Statistics

Message	GST		
Description	GNSS Pseudo Range Error Statistics		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	This message reports statistical information on the quality of the position solution.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x07	11	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxGST	-	string	\$GPGST	GST Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	082356.00	UTC time of associated position fix, see note on UTC representation
2	rangeRms	m	numeric	1.8	RMS value of the standard deviation of the ranges
3	stdMajor	m	numeric	-	Standard deviation of semi-major axis (blank - not supported)
4	stdMinor	m	numeric	-	Standard deviation of semi-minor axis (blank - not supported)
5	orient	deg	numeric	-	Orientation of semi-major axis (blank - not supported)
6	stdLat	m	numeric	1.7	Standard deviation of latitude error
7	stdLong	m	numeric	1.3	Standard deviation of longitude error
8	stdAlt	m	numeric	2.2	Standard deviation of altitude error
9	cs	-	hexadecimal	*7E	Checksum
10	<CR><LF>	-	character	-	Carriage return and line feed

18.13 GSV

18.13.1 GNSS Satellites in View

Message	GSV		
Description	GNSS Satellites in View		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	The number of satellites in view, together with each SV ID, elevation azimuth, and signal strength (C/No) value. Only four satellite details are transmitted in one message. In a multi-GNSS system sets of GSV messages will be output multiple times, one set for each GNSS.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x03	8..16	

Message Structure:

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

Example:

```
$GPGSV,3,1,10,23,38,230,44,29,71,156,47,07,29,116,41,08,09,081,36,0*7F
```

```
$GPGSV,3,2,10,10,07,189,,05,05,220,,09,34,274,42,18,25,309,44,0*72
```

```
$GPGSV,3,3,10,26,82,187,47,28,43,056,46,0*77
```

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

18.14 RMC

18.14.1 Recommended Minimum data

Message	RMC		
Description	Recommended Minimum data		
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 		
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 GNSS system data.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x04	16	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxRMC	-	string	\$GPRMC	RMC Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	083559.00	UTC time, see note on UTC representation
2	status	-	character	A	Status, V = Navigation receiver warning, A = Data valid, see position fix flags description
3	lat	-	ddmm. mmmm	4717.11437	Latitude (degrees & minutes), see format description
4	NS	-	character	N	North/South indicator
5	long	-	dddmm. mmmm	00833.91522	Longitude (degrees & minutes), see format description
6	EW	-	character	E	East/West indicator
7	spd	knot s	numeric	0.004	Speed over ground
8	cog	degr ees	numeric	77.52	Course over ground
9	date	-	ddmmyy	091202	Date in day, month, year format, see note on UTC representation
10	mv	degr ees	numeric	-	Magnetic variation value (blank - not supported)
11	mvEW	-	character	-	Magnetic variation E/W indicator (blank - not supported)
12	posMode	-	character	-	Mode Indicator, see position fix flags description NMEA v2.3 and above only
13	navStatus	-	character	V	Navigational status indicator (V = Equipment is not providing navigational status information) NMEA v4.1 and above only
14	cs	-	hexadecimal	*57	Checksum
15	<CR><LF>	-	character	-	Carriage return and line feed

18.15 TXT

18.15.1 Text Transmission

Message	TXT		
Description	Text Transmission		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	This message is not configured through UBX-CFG-MSG, but instead through UBX-CFG-INF. This message outputs various information on the receiver, such as power-up screen, software version etc. This message can be configured using UBX Protocol message UBX-CFG-INF .		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x41	7	

Message Structure:

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

Example:

```
$GPTXT,01,01,02,u-blox ag - www.u-blox.com*50
```

```
$GPTXT,01,01,02,ANTARIS ATR0620 HW 00000040*67
```

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

18.16 VLW

18.16.1 Dual ground/water distance

Message	VLW		
Description	Dual ground/water distance		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	The distance traveled, relative to the water and over the ground.		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x0F	11	

Message Structure:

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

Example:

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

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

18.17 VTG

18.17.1 Course over ground and Ground speed

Message	VTG		
Description	Course over ground and Ground speed		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
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:

```
$xxVTG, cogt, T, ,M, 0.004, N, 0.008, K, posMode *cs<CR><LF>
```

Example:

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

Field No.	Name	Unit	Format	Example	Description
-----------	------	------	--------	---------	-------------

VTG continued

Field No.	Name	Unit	Format	Example	Description
0	xxVTG	-	string	\$GPVTG	VTG Message ID (xx = current Talker ID)
1	cogt	degrees	numeric	77.52	Course over ground (true)
2	T	-	character	T	Fixed field: true
3	cogm	degrees	numeric	-	Course over ground (magnetic), not output
4	M	-	character	M	Fixed field: magnetic
5	knots	knots	numeric	0.004	Speed over ground
6	N	-	character	N	Fixed field: knots
7	kph	km/h	numeric	0.008	Speed over ground
8	K	-	character	K	Fixed field: kilometers per hour
9	posMode	-	character	A	Mode Indicator, see position fix flags description NMEA v2.3 and above only
10	cs	-	hexadecimal	*06	Checksum
11	<CR><LF>	-	character	-	Carriage return and line feed

18.18 ZDA

18.18.1 Time and Date

Message	ZDA		
Description	Time and Date		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
Type	Output Message		
Comment	-		
Message Info	ID for CFG-MSG	Number of fields	
	0xF0 0x08	9	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	xxZDA	-	string	\$GPZDA	ZDA Message ID (xx = current Talker ID)
1	time	-	hhmmss.ss	082710.00	UTC Time, see note on UTC representation
2	day	day	dd	16	UTC day (range: 1-31)
3	month	month	mm	09	UTC month (range: 1-12)
4	year	year	yyyy	2002	UTC year
5	ltzh	-	-xx	00	Local time zone hours (fixed to 00)
6	ltzn	-	zz	00	Local time zone minutes (fixed to 00)
7	cs	-	hexadecimal	*64	Checksum

ZDA continued

<i>Field No.</i>	<i>Name</i>	<i>Unit</i>	<i>Format</i>	<i>Example</i>	<i>Description</i>
8	<CR><LF>	-	character	-	Carriage return and line feed

19 PUBX Messages

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

19.1 CONFIG (PUBX,41)

19.1.1 Set Protocols and Baudrate

Message	CONFIG		
Description	Set Protocols and Baudrate		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
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.	Name	Unit	Format	Example	Description
0	\$PUBX	-	string	\$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	41	Proprietary message identifier
2	portId	-	numeric	1	ID of communication port. For a list of port IDs see Serial Communication Ports Description .
3	inProto	-	hexadecimal	0007	Input protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in UBX-CFG-PRT .
4	outProto	-	hexadecimal	0003	Output protocol mask. Bitmask, specifying which protocols(s) are allowed for input. For details see corresponding field in UBX-CFG-PRT .
5	baudrate	bits/s	numeric	19200	Baudrate
6	autobauding	-	numeric	0	Autobauding: 1=enable, 0=disable (not supported on u-blox 5, set to 0)
7	cs	-	hexadecimal	*25	Checksum
8	<CR><LF>	-	character	-	Carriage return and line feed

19.2 POSITION (PUBX,00)

19.2.1 Lat/Long Position Data

Message	POSITION		
Description	Lat/Long Position Data		
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
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 UBX-CFG-DAT .		
Message Info	ID for CFG-MSG	Number of fields	
	0xF1 0x00	23	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	\$PUBX	-	string	\$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	00	Proprietary message identifier: 00
2	time	-	hhmmss.ss	081350.00	UTC time, see note on UTC representation
3	lat	-	ddmm.mmmmm	4717.113210	Latitude (degrees & minutes), see format description
4	NS	-	character	N	North/South Indicator
5	long	-	dddmm.mmmmm	00833.915187	Longitude (degrees & minutes), see format description
6	EW	-	character	E	East/West indicator
7	altRef	m	numeric	546.589	Altitude above user datum ellipsoid.
8	navStat	-	string	G3	Navigation Status, See Table below
9	hAcc	m	numeric	2.1	Horizontal accuracy estimate.
10	vAcc	m	numeric	2.0	Vertical accuracy estimate.
11	SOG	km/h	numeric	0.007	Speed over ground
12	COG	deg	numeric	77.52	Course over ground
13	vVel	m/s	numeric	0.007	Vertical velocity (positive downwards)
14	diffAge	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
15	HDOP	-	numeric	0.92	HDOP, Horizontal Dilution of Precision
16	VDOP	-	numeric	1.19	VDOP, Vertical Dilution of Precision
17	TDOP	-	numeric	0.77	TDOP, Time Dilution of Precision
18	numSvs	-	numeric	9	Number of satellites used in the navigation solution
19	reserved	-	numeric	0	Reserved, always set to 0

POSITION continued

Field No.	Name	Unit	Format	Example	Description
20	DR	-	numeric	0	DR used
21	cs	-	hexadecimal	*5B	Checksum
22	<CR><LF>	-	character	-	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

19.3 SVSTATUS (PUBX,03)
19.3.1 Satellite Status

Message	SVSTATUS		
Description	Satellite Status		
Firmware	Supported on: • u-blox M8 firmware version 2.00		
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*n	

Message Structure:

```
$PUBX,03,GT{,sv,s,az,e1,cno,lck},*cs<CR><LF>
```

Example:

```
$PUBX,03,11,23,-,-,45,010,29,-,-,46,013,07,-,-,42,015,08,U,067,31,42,025,10,U,195,33,46,026,18,U,326,08,39,026,17,-,-,32,015,26,U,306,66,48,025,27,U,073,10,36,026,28,U,089,61,46,024,15,-,-,39,014*0D
```

Field No.	Name	Unit	Format	Example	Description
0	\$PUBX	-	string	\$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	03	Proprietary message identifier: 03
2	n	-	numeric	11	Number of GPS satellites tracked
<i>Start of repeated block (n times)</i>					
3 + 6*N	sv	-	numeric	23	Satellite ID
4 + 6*N	s	-	character	-	Satellite status, see table below
5 + 6*N	az	deg	numeric	-	Satellite azimuth (range: 0-359)

SVSTATUS continued

Field No.	Name	Unit	Format	Example	Description
6 + 6*N	e1	deg	numeric	-	Satellite elevation (range: 0-90)
7 + 6*N	cno	dBH z	numeric	45	Signal strength (C/N0, range 0-99), blank when not tracking
8 + 6*N	lck	s	numeric	010	Satellite carrier lock time (range: 0-64) 0: code lock only 64: lock for 64 seconds or more
<i>End of repeated block</i>					
3 + 6*n	cs	-	hexadecimal	*0D	Checksum
4 + 6*n	<CR><LF>	-	character	-	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

19.4 TIME (PUBX,04)

19.4.1 Time of Day and Clock Information

<i>Message</i>	TIME		
<i>Description</i>	Time of Day and Clock Information		
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 		
<i>Type</i>	Output Message		
<i>Comment</i>	-		
<i>Message Info</i>	<i>ID for CFG-MSG</i>	<i>Number of fields</i>	
	0xF1 0x04	12	

Message Structure:

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

Example:

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

Field No.	Name	Unit	Format	Example	Description
0	\$PUBX	-	string	\$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	04	Proprietary message identifier: 04
2	time	-	hhmmss.ss	073731.00	UTC time, see note on UTC representation
3	date	-	ddmmyy	091202	UTC date, day, month, year format, see note on UTC representation
4	utcTow	s	numeric	113851.00	UTC Time of Week
5	utcWk	-	numeric	1196	UTC week number, continues beyond 1023

TIME continued

<i>Field No.</i>	<i>Name</i>	<i>Unit</i>	<i>Format</i>	<i>Example</i>	<i>Description</i>
6	leapSec	s	numeric/text	15D	Leap seconds The number is marked with a 'D' if the value is the firmware default value. If the value is not marked it has been received from a satellite.
7	clkBias	ns	numeric	1930035	Receiver clock bias
8	clkDrift	ns/s	numeric	-2660.664	Receiver clock drift
9	tpGran	ns	numeric	43	Time Pulse Granularity, The quantization error of the TIMEPULSE pin
10	cs	-	hexadecimal	*3C	Checksum
11	<CR><LF>	-	character	-	Carriage Return and Line Feed

UBX Protocol

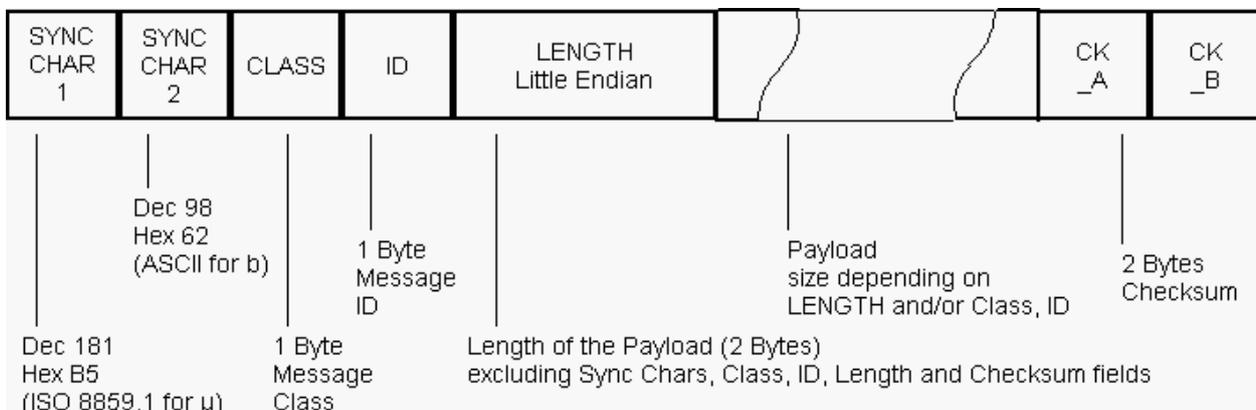
20 UBX Protocol Key Features

u-blox GNSS receivers use a u-blox proprietary protocol to transmit GNSS data to a host computer. This protocol has the following key features:

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

21 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.

22 UBX Payload Definition Rules

22.1 Structure Packing

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

22.2 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

22.3 Number Formats

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

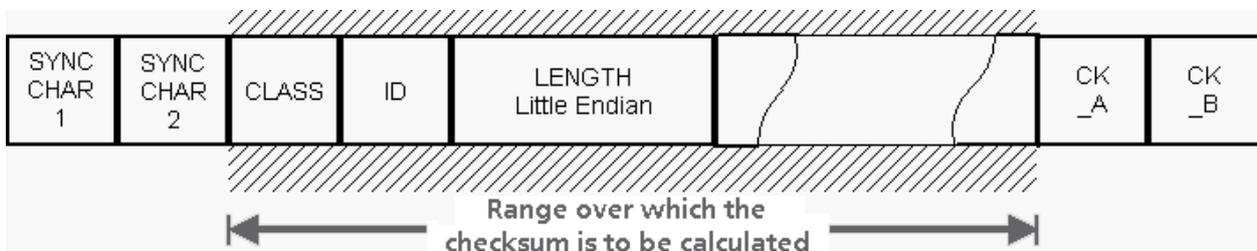
All floating point values are transmitted in IEEE754 single or double precision. A technical description of the IEEE754 format can be found in the AnswerBook from the ADS1.x toolkit.

Variable Type Definitions

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

23 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.

24 UBX Message Flow

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

24.1 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.

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

24.2 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.

25 UBX Satellite Numbering

UBX protocol messages use two different numbering schemes. Many UBX messages (e.g. [UBX-NAV-SVINFO](#)) use a single byte for the satellite identifier (normally named "svId"). This uses numbering similar to the "extended" NMEA scheme and is merely an extension of the scheme in use for previous generations of u-blox receivers.

With ever increasing numbers of GNSS satellites, this scheme will have to be phased out in future u-blox receivers (as numbers greater than 255 will become necessary). Consequently, newer messages use a more sophisticated, flexible and future-proof approach. This involves having a separate *gnssId* to identify which GNSS type the satellite is part of and a simple *svId* which indicates which number the satellite is in that system. In nearly all cases, this means that the "svId" is the natural number associated with the satellite. For example the GLONASS SV4 is identified as *gnssId* 6, *svId* 4, while the GPS SV4 is *gnssId* 0, *svId* 4.

See [Satellite Numbering Summary](#) for a complete list of satellite numbers.

GNSS Identifiers

<i>gnssId</i>	<i>GNSS Type</i>
0	GPS
1	SBAS
2	Galileo
3	BeiDou
5	QZSS
6	GLONASS

Other values will be added as support for other GNSS types is enabled in u-blox receivers.



GLONASS satellites can be tracked before they have been identified. In UBX messages, such unknown satellite numbers are always reported with *svid* 255.

26 UBX Class IDs

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

<i>Name</i>	<i>Class</i>	<i>Description</i>
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.
UPD	0x09	Firmware Update Messages: Memory/Flash erase/write, Reboot, Flash identification, etc.
MON	0x0A	Monitoring Messages: Communication Status, CPU Load, Stack Usage, Task Status
AID	0x0B	AssistNow Aiding Messages: Ephemeris, Almanac, other A-GPS data input
TIM	0x0D	Timing Messages: Time Pulse Output, Timemark Results
MGA	0x13	Multi-GNSS Assistance: Assistance data for various GNSS
LOG	0x21	Logging Messages: Log creation, deletion, info and retrieval

All remaining class IDs are reserved.

27 UBX Messages Overview

Page	Mnemonic	Cls/ID	Length	Type	Description
UBX Class ACK				Ack/Nack Messages	
89	ACK-ACK	0x05 0x01	2	Output	Message Acknowledged
89	ACK-NAK	0x05 0x00	2	Output	Message Not-Acknowledged
UBX Class AID				AssistNow Aiding Messages	
90	AID-ALM	0x0B 0x30	0	Poll Request	Poll GPS Aiding Almanac Data
90	AID-ALM	0x0B 0x30	1	Poll Request	Poll GPS Aiding Almanac Data for a SV
91	AID-ALM	0x0B 0x30	(8) or (40)	Input/Output	GPS Aiding Almanac Input/Output Message
91	AID-AOP	0x0B 0x33	0	Poll request	Poll AssistNow Autonomous data, all GPS satellites
92	AID-AOP	0x0B 0x33	1	Poll request	Poll AssistNow Autonomous data, one GPS satellite
92	AID-AOP	0x0B 0x33	68	Input/Output	AssistNow Autonomous data
93	AID-EPH	0x0B 0x31	0	Poll Request	Poll GPS Aiding Ephemeris Data
93	AID-EPH	0x0B 0x31	1	Poll Request	Poll GPS Aiding Ephemeris Data for a SV
94	AID-EPH	0x0B 0x31	(8) or (104)	Input/Output	GPS Aiding Ephemeris Input/Output Message
95	AID-HUI	0x0B 0x02	0	Poll Request	Poll GPS Health, UTC and ionosphere parameters
95	AID-HUI	0x0B 0x02	72	Input/Output	GPS Health, UTC and ionosphere parameters
96	AID-INI	0x0B 0x01	0	Poll Request	Poll GPS Initial Aiding Data
97	AID-INI	0x0B 0x01	48	Input/Output	Aiding position, time, frequency, clock drift
UBX Class CFG				Configuration Input Messages	
99	CFG-ANT	0x06 0x13	0	Poll Request	Poll Antenna Control Settings
99	CFG-ANT	0x06 0x13	4	Input/Output	Antenna Control Settings
100	CFG-CFG	0x06 0x09	(12) or (13)	Command	Clear, Save and Load configurations
102	CFG-DAT	0x06 0x06	0	Poll Request	Poll Datum Setting
102	CFG-DAT	0x06 0x06	44	Input	Set User-defined Datum
103	CFG-DAT	0x06 0x06	52	Output	The currently defined Datum
104	CFG-GNSS	0x06 0x3E	0	Poll Request	Poll the GNSS system configuration
104	CFG-GNSS	0x06 0x3E	4 + 8*numConfigBlocks	Input/Output	GNSS system configuration
105	CFG-INF	0x06 0x02	1	Poll Request	Poll INF message configuration for one protocol
106	CFG-INF	0x06 0x02	0 + 10*N	Input/Output	Information message configuration
107	CFG-ITFM	0x06 0x39	0	Poll Request	Poll Jamming/Interference Monitor configuration.
107	CFG-ITFM	0x06 0x39	8	Command	Jamming/Interference Monitor configuration.
108	CFG-LOGFILTER	0x06 0x47	0	Poll Request	Poll Data Logger filter Configuration
108	CFG-LOGFILTER	0x06 0x47	12	Input/Output	Data Logger Configuration
110	CFG-MSG	0x06 0x01	2	Poll Request	Poll a message configuration
110	CFG-MSG	0x06 0x01	8	Input/Output	Set Message Rate(s)
111	CFG-MSG	0x06 0x01	3	Input/Output	Set Message Rate
111	CFG-NAV5	0x06 0x24	0	Poll Request	Poll Navigation Engine Settings
111	CFG-NAV5	0x06 0x24	36	Input/Output	Navigation Engine Settings

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
113	CFG-NAVX5	0x06 0x23	0	Poll Request	Poll Navigation Engine Expert Settings
113	CFG-NAVX5	0x06 0x23	40	Input/Output	Navigation Engine Expert Settings
115	CFG-NMEA	0x06 0x17	0	Poll Request	Poll the NMEA protocol configuration
115	CFG-NMEA	0x06 0x17	4	Input/Output	NMEA protocol configuration (deprecated)
117	CFG-NMEA	0x06 0x17	12	Input/Output	NMEA protocol configuration V0 (deprecated)
119	CFG-NMEA	0x06 0x17	20	Input/Output	Extended NMEA protocol configuration V1
122	CFG-ODO	0x06 0x1E	0	Poll Request	Poll Odometer, Low-speed COG Engine Settings
122	CFG-ODO	0x06 0x1E	20	Input/Output	Odometer, Low-speed COG Engine Settings
123	CFG-PM2	0x06 0x3B	0	Poll Request	Poll extended Power Management configuration
124	CFG-PM2	0x06 0x3B	44	Input/Output	Extended Power Management configuration
125	CFG-PRT	0x06 0x00	0	Poll Request	Polls the configuration of the used I/O Port
126	CFG-PRT	0x06 0x00	1	Poll Request	Polls the configuration for one I/O Port
126	CFG-PRT	0x06 0x00	20	Input/Output	Port Configuration for UART
129	CFG-PRT	0x06 0x00	20	Input/Output	Port Configuration for USB Port
131	CFG-PRT	0x06 0x00	20	Input/Output	Port Configuration for SPI Port
133	CFG-PRT	0x06 0x00	20	Input/Output	Port Configuration for DDC Port
136	CFG-PWR	0x06 0x57	8	Set	Put the receiver in a defined power state
136	CFG-RATE	0x06 0x08	0	Poll Request	Poll Navigation/Measurement Rate Settings
137	CFG-RATE	0x06 0x08	6	Input/Output	Navigation/Measurement Rate Settings
137	CFG-RINV	0x06 0x34	0	Poll Request	Poll contents of Remote Inventory
138	CFG-RINV	0x06 0x34	1 + 1*N	Input/Output	Contents of Remote Inventory
138	CFG-RST	0x06 0x04	4	Command	Reset Receiver / Clear Backup Data Structures
140	CFG-RXM	0x06 0x11	0	Poll Request	Poll RXM configuration
140	CFG-SBAS	0x06 0x16	0	Poll Request	Poll contents of SBAS Configuration
140	CFG-SBAS	0x06 0x16	8	Input/Output	SBAS Configuration
142	CFG-TP5	0x06 0x31	0	Poll Request	Poll Time Pulse Parameters
142	CFG-TP5	0x06 0x31	1	Poll Request	Poll Time Pulse Parameters
143	CFG-TP5	0x06 0x31	32	Input/Output	Time Pulse Parameters
144	CFG-USB	0x06 0x1B	0	Poll Request	Poll a USB configuration
144	CFG-USB	0x06 0x1B	108	Input/Output	USB Configuration
UBX Class INF				Information Messages	
146	INF-DEBUG	0x04 0x04	0 + 1*N	Output	ASCII String output, indicating debug output
146	INF-ERROR	0x04 0x00	0 + 1*N	Output	ASCII String output, indicating an error
147	INF-NOTICE	0x04 0x02	0 + 1*N	Output	ASCII String output, with informational contents
147	INF-TEST	0x04 0x03	0 + 1*N	Output	ASCII String output, indicating test output
148	INF-WARNING	0x04 0x01	0 + 1*N	Output	ASCII String output, indicating a warning
UBX Class LOG				Logging Messages	
149	LOG-CREATE	0x21 0x07	8	Command	Create Log File

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
150	LOG-ERASE	0x21 0x03	0	Command	Erase Logged Data
150	LOG-FINDTIME	0x21 0x0E	12	Input	Find index of the first log entry <= given time
151	LOG-FINDTIME	0x21 0x0E	8	Output	Response to FINDTIME request.
151	LOG-INFO	0x21 0x08	0	Poll Request	Poll for log information
151	LOG-INFO	0x21 0x08	48	Output	Log information
153	LOG-RETRIEVEPOSEXTIME	0x21 0x0f	32	Output	Odometer log entry
154	LOG-RETRIEVEPOS	0x21 0x0b	40	Output	Position fix log entry
155	LOG-RETRIEVESTRING	0x21 0x0d	16 + 1*byteCount	Output	Byte string log entry
155	LOG-RETRIEVE	0x21 0x09	12	Command	Request log data
156	LOG-STRING	0x21 0x04	0 + 1*N	Command	Store arbitrary string in on-board Flash memory
UBX Class MGA				Multi-GNSS Assistance	
157	MGA-ACK-DATA0	0x13 0x60	8	Output	Multi-GNSS Acknowledge message
158	MGA-ANO	0x13 0x20	76	Input	Multi-GNSS AssistNow Offline Assistance
158	MGA-DBD	0x13 0x80	0	Poll Request	Poll the Navigation Database.
159	MGA-DBD	0x13 0x80	12 + 1*N	Input / Output Message	Navigation Database Dump Entry
159	MGA-FLASH-DATA	0x13 0x21	6 + 1*size	Input	Transfer MGA-ANO data block to flash.
160	MGA-FLASH-STOP	0x13 0x21	2	Input	Finish flashing MGA-ANO data.
160	MGA-FLASH-ACK	0x13 0x21	6	Output	Acknowledge last FLASH-DATA or FLASH-STOP.
161	MGA-GLO-EPH	0x13 0x06	48	Input	GLONASS Ephemeris Assistance
162	MGA-GLO-ALM	0x13 0x06	36	Input	GLONASS Almanac Assistance
163	MGA-GLO-TIMEOFFSET	0x13 0x06	20	Input	GLONASS Auxiliary Time Offset Assistance
164	MGA-GPS-EPH	0x13 0x00	68	Input	GPS Ephemeris Assistance
165	MGA-GPS-ALM	0x13 0x00	36	Input	GPS Almanac Assistance
166	MGA-GPS-HEALTH	0x13 0x00	40	Input	GPS Health Assistance
167	MGA-GPS-UTC	0x13 0x00	20	Input	GPS UTC Assistance
167	MGA-GPS-IONO	0x13 0x00	16	Input	GPS Ionosphere Assistance
168	MGA-INI-POS_XYZ	0x13 0x40	20	Input	Initial Position Assistance
169	MGA-INI-POS_LLH	0x13 0x40	20	Input	Initial Position Assistance
170	MGA-INI-TIME_UTC	0x13 0x40	24	Input	Initial Time Assistance
171	MGA-INI-TIME_GNSS	0x13 0x40	24	Input	Initial Time Assistance
172	MGA-INI-CLKD	0x13 0x40	12	Input	Initial Clock Drift Assistance
173	MGA-INI-FREQ	0x13 0x40	12	Input	Initial Frequency Assistance
174	MGA-INI-EOP	0x13 0x40	72	Input	Earth Orientation Parameters Assistance
174	MGA-QZSS-EPH	0x13 0x05	68	Input	QZSS Ephemeris Assistance
176	MGA-QZSS-ALM	0x13 0x05	36	Input	QZSS Almanac Assistance
177	MGA-QZSS-HEALTH	0x13 0x05	12	Input	QZSS Health Assistance
UBX Class MON				Monitoring Messages	
178	MON-GNSS	0x0A 0x28	8	Output	Information message GNSS selection

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
179	MON-HW2	0x0A 0x0B	28	Periodic/Polled	Extended Hardware Status
180	MON-HW	0x0A 0x09	60	Periodic/Polled	Hardware Status
182	MON-IO	0x0A 0x02	0 + 20*N	Periodic/Polled	I/O Subsystem Status
182	MON-MSGPP	0x0A 0x06	120	Periodic/Polled	Message Parse and Process Status
183	MON-PATCH	0x0A 0x27	0	Poll Request	Poll Request for installed patches
183	MON-PATCH	0x0A 0x27	4 + 16*nEntries	Output Message	Output information about installed patches.
184	MON-RXBUF	0x0A 0x07	24	Periodic/Polled	Receiver Buffer Status
185	MON-RXR	0x0A 0x21	1	Output	Receiver Status Information
185	MON-TXBUF	0x0A 0x08	28	Periodic/Polled	Transmitter Buffer Status
186	MON-VER	0x0A 0x04	0	Poll Request	Poll Receiver/Software Version
187	MON-VER	0x0A 0x04	40 + 30*N	Answer to Poll	Receiver/Software Version
UBX Class NAV				Navigation Results	
188	NAV-AOPSTATUS	0x01 0x60	16	Periodic/Polled	AssistNow Autonomous Status
189	NAV-CLOCK	0x01 0x22	20	Periodic/Polled	Clock Solution
189	NAV-DGPS	0x01 0x31	16 + 12*numCh	Periodic/Polled	DGPS Data Used for NAV
190	NAV-DOP	0x01 0x04	18	Periodic/Polled	Dilution of precision
191	NAV-ODO	0x01 0x09	20	Periodic/Polled	Odometer Solution
191	NAV-ORB	0x01 0x34	8 + 6*numSv	Periodic/Polled	GNSS Orbit Database Info
194	NAV-POSECEF	0x01 0x01	20	Periodic/Polled	Position Solution in ECEF
195	NAV-POSLLH	0x01 0x02	28	Periodic/Polled	Geodetic Position Solution
195	NAV-PVT	0x01 0x07	92	Periodic/Polled	Navigation Position Velocity Time Solution
198	NAV-RESETODO	0x01 0x10	0	Command	Reset odometer
198	NAV-SAT	0x01 0x35	8 + 12*numSvs	Periodic/Polled	Satellite Information
200	NAV-SBAS	0x01 0x32	12 + 12*cnt	Periodic/Polled	SBAS Status Data
201	NAV-SOL	0x01 0x06	52	Periodic/Polled	Navigation Solution Information
202	NAV-STATUS	0x01 0x03	16	Periodic/Polled	Receiver Navigation Status
204	NAV-SVINFO	0x01 0x30	8 + 12*numCh	Periodic/Polled	Space Vehicle Information
206	NAV-TIMEGPS	0x01 0x20	16	Periodic/Polled	GPS Time Solution
207	NAV-TIMEUTC	0x01 0x21	20	Periodic/Polled	UTC Time Solution
208	NAV-VELECEF	0x01 0x11	20	Periodic/Polled	Velocity Solution in ECEF
209	NAV-VELNED	0x01 0x12	36	Periodic/Polled	Velocity Solution in NED
UBX Class RXM				Receiver Manager Messages	
210	RXM-PMREQ	0x02 0x41	8	Command	Requests a Power Management task
210	RXM-SVSI	0x02 0x20	8 + 6*numSV	Periodic/Polled	SV Status Info
UBX Class TIM				Timing Messages	
213	TIM-TM2	0x0D 0x03	28	Periodic/Polled	Time mark data
214	TIM-TP	0x0D 0x01	16	Periodic/Polled	Time Pulse Timedata
215	TIM-VRFY	0x0D 0x06	20	Polled/Once	Sourced Time Verification

UBX Messages Overview continued

Page	Mnemonic	Cls/ID	Length	Type	Description
UBX Class UPD			Firmware Update Messages		
217	UPD-SOS	0x09 0x14	0	Poll Request	Poll Backup File Restore Status
217	UPD-SOS	0x09 0x14	4	Input	Create Backup File in Flash
218	UPD-SOS	0x09 0x14	4	Input	Clear Backup in Flash
218	UPD-SOS	0x09 0x14	8	Output	Backup File Creation Acknowledge
219	UPD-SOS	0x09 0x14	8	Output	System Restored from Backup

28 UBX-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.

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

28.1.1 Message Acknowledged

Message	ACK-ACK					
Description	Message Acknowledged					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	Output upon processing of an input message					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x05	0x01	2	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	clsID	-	Class ID of the Acknowledged Message	
1	U1	-	msgID	-	Message ID of the Acknowledged Message	

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

28.2.1 Message Not-Acknowledged

Message	ACK-NAK					
Description	Message Not-Acknowledged					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	Output upon processing of an input message					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x05	0x00	2	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	clsID	-	Class ID of the Not-Acknowledged Message	
1	U1	-	msgID	-	Message ID of the Not-Acknowledged Message	

29 UBX-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.

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

29.1.1 Poll GPS Aiding Almanac Data

Message	AID-ALM					
Description	Poll GPS Aiding Almanac Data					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead Poll GPS Aiding Data (Almanac) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type AID-ALM as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x30	0	see below	CK_A CK_B
No payload						

29.1.2 Poll GPS Aiding Almanac Data for a SV

Message	AID-ALM					
Description	Poll GPS Aiding Almanac Data for a SV					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead Poll GPS Aiding Data (Almanac) for an SV by sending this message to the receiver. The receiver will return one message of type AID-ALM as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x30	1	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	svid	-	SV ID for which the receiver shall return its Almanac Data (Valid Range: 1 .. 32 or 51, 56, 63).	

29.1.3 GPS Aiding Almanac Input/Output Message

Message	AID-ALM					
Description	GPS Aiding Almanac Input/Output Message					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead <ul style="list-style-type: none"> • 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	Class	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]	-	dwrđ	-	Almanac Words	
End of optional block						

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

29.2.1 Poll AssistNow Autonomous data, all GPS satellites

Message	AID-AOP					
Description	Poll AssistNow Autonomous data, all GPS satellites					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Poll request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead <p>Poll AssistNow Autonomous aiding data for all GPS satellites by sending this empty message. The receiver will return an AID-AOP message (see definition below) for each GPS satellite for which data is available.</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x33	0	see below	CK_A CK_B

No payload

29.2.2 Poll AssistNow Autonomous data, one GPS satellite

Message	AID-AOP					
Description	Poll AssistNow Autonomous data, one GPS satellite					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Poll request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead Poll the <i>AssistNow Autonomous</i> data for the specified GPS satellite. The receiver will return a AID-AOP message (see definition below) if data is available for the requested satellite.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x33	1	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	svid	-	GPS SV ID for which the data is requested (valid range: 1..32).	

29.2.3 AssistNow Autonomous data

Message	AID-AOP					
Description	AssistNow Autonomous data					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead If enabled, this message is output at irregular intervals. It is output whenever <i>AssistNow Autonomous</i> 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 <i>AssistNow Autonomous</i> feature on the receiver. See the section AssistNow Autonomous in the receiver description for details on this feature.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x33	68	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	gnssId	-	GNSS identifier (see Satellite Numbering)	
1	U1	-	svId	-	Satellite identifier (see Satellite Numbering)	
2	U2	-	reserved	-	reserved	
4	U1[64]	-	data	-	assistance data	

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

29.3.1 Poll GPS Aiding Ephemeris Data

<i>Message</i>	AID-EPH					
<i>Description</i>	Poll GPS Aiding Ephemeris Data					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	All UBX-AID messages are deprecated; use UBX-MGA messages instead Poll GPS Aiding Data (Ephemeris) for all 32 SVs by sending this message to the receiver without any payload. The receiver will return 32 messages of type AID-EPH as defined below.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x0B	0x31	0	see below	CK_A CK_B
<i>No payload</i>						

29.3.2 Poll GPS Aiding Ephemeris Data for a SV

<i>Message</i>	AID-EPH					
<i>Description</i>	Poll GPS Aiding Ephemeris Data for a SV					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	All UBX-AID messages are deprecated; use UBX-MGA messages instead Poll GPS Constellation Data (Ephemeris) for an SV by sending this message to the receiver. The receiver will return one message of type AID-EPH as defined below.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x0B	0x31	1	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	svid	-	SV ID for which the receiver shall return its Ephemeris Data (Valid Range: 1 .. 32).	

29.3.3 GPS Aiding Ephemeris Input/Output Message

Message	AID-EPH					
Description	GPS Aiding Ephemeris Input/Output Message					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	<p>All UBX-AID messages are deprecated; use UBX-MGA messages instead</p> <ul style="list-style-type: none"> • 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	Class	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						

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

29.4.1 Poll GPS Health, UTC and ionosphere parameters

Message	AID-HUI					
Description	Poll GPS Health, UTC and ionosphere parameters					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x02	0	see below	CK_A CK_B
No payload						

29.4.2 GPS Health, UTC and ionosphere parameters

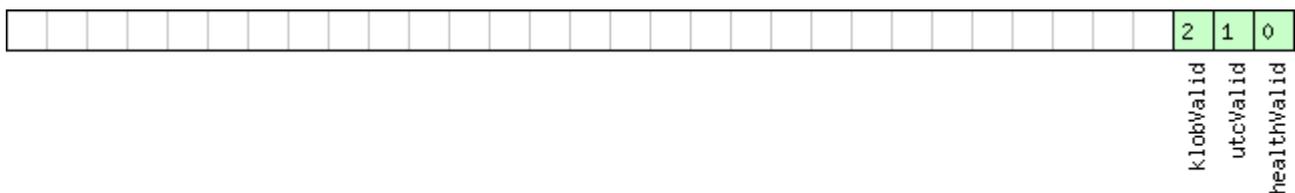
Message	AID-HUI					
Description	GPS Health, UTC and ionosphere parameters					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead This message contains a health bit mask, UTC time and Klobuchar parameters. For more information on these parameters, see the ICD-GPS-200 documentation.					
Message Structure	Header	Class	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 representst 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	

AID-HUI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
44	R4	-	klobA2	s/semicircle ²	Klobuchar - alpha 2
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



- signed value
- unsigned value
- reserved

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

29.5 UBX-AID-INI (0x0B 0x01)
29.5.1 Poll GPS Initial Aiding Data

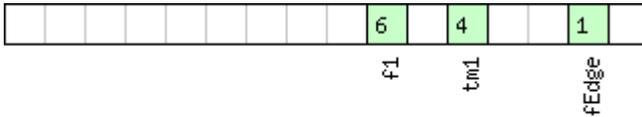
Message	AID-INI					
Description	Poll GPS Initial Aiding Data					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x01	0	see below	CK_A CK_B
No payload						

29.5.2 Aiding position, time, frequency, clock drift

Message	AID-INI					
Description	Aiding position, time, frequency, clock drift					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	All UBX-AID messages are deprecated; use UBX-MGA messages instead This message contains position, time and clock drift information. The position can be input in either the ECEF X/Y/Z coordinate system or as lat/lon/height. The time can either be input as inexact value via the standard communication interface, suffering from latency depending on the baud rate, or using hardware time synchronization where an accurate time pulse is input on the external interrupts. It is also possible to supply hardware frequency aiding by connecting a continuous signal to an external interrupt.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0B	0x01	48	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	I4	-	ecefXOrLat	cm_or_deg*1e-7	WGS84 ECEF X coordinate or latitude, depending on flags below	
4	I4	-	ecefYOrLon	cm_or_deg*1e-7	WGS84 ECEF Y coordinate or longitude, depending on flags below	
8	I4	-	ecefZOrAlt	cm	WGS84 ECEF Z coordinate or altitude, depending on flags below	
12	U4	-	posAcc	cm	Position accuracy (stddev)	
16	X2	-	tmCfg	-	Time mark configuration (see graphic below)	
18	U2	-	wnoOrDate	week_or_year Month	Actual week number or yearSince2000/Month (YYMM), depending on flags below	
20	U4	-	towOrTime	ms_or_dayHourMinute Sec	Actual time of week or DayOfMonth/Hour/Minute/Second (DDHHMMSS), depending on flags below	
24	I4	-	towNs	ns	Fractional part of time of week	
28	U4	-	tAccMs	ms	Milliseconds part of time accuracy	
32	U4	-	tAccNs	ns	Nanoseconds part of time accuracy	
36	I4	-	clkDOrFreq	ns/s_or_Hz*1e-2	Clock drift or frequency, depending on flags below	
40	U4	-	clkDAccOrFreqAcc	ns/s_or_ppb	Accuracy of clock drift or frequency, depending on flags below	
44	X4	-	flags	-	Bitmask with the following flags (see graphic below)	

Bitfield tmCfg

This Graphic explains the bits of tmCfg

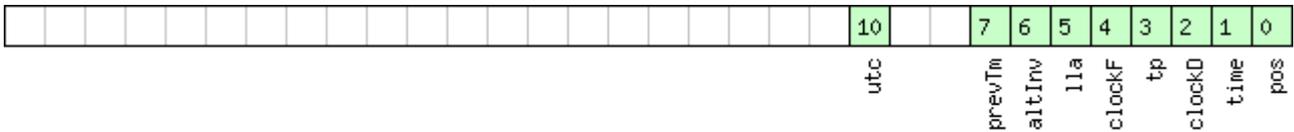


- signed value
- unsigned value
- reserved

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



- signed value
- unsigned value
- reserved

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

30 UBX-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.

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

30.1.1 Poll Antenna Control Settings

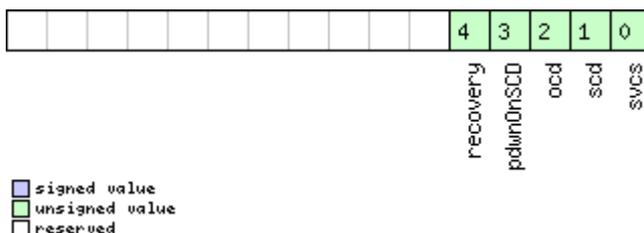
Message	CFG-ANT					
Description	Poll Antenna Control Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-ANT with a payload as defined below					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x13	0	see below	CK_A CK_B
No payload						

30.1.2 Antenna Control Settings

Message	CFG-ANT					
Description	Antenna Control Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x13	4	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	X2	-	flags	-	Antenna Flag Mask (see graphic below)	
2	X2	-	pins	-	Antenna Pin Configuration (see graphic below)	

Bitfield flags

This Graphic explains the bits of flags



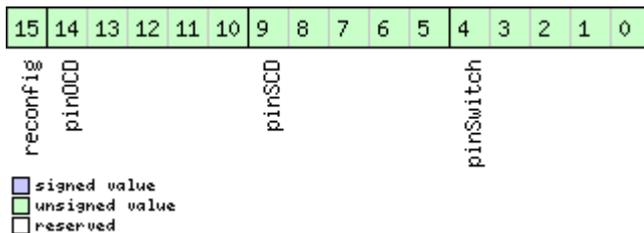
Name	Description
svcs	Enable Antenna Supply Voltage Control Signal
scd	Enable Short Circuit Detection

Bitfield flags Description continued

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

Bitfield pins

This Graphic explains the bits of pins



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

30.2 UBX-CFG-CFG (0x06 0x09)
30.2.1 Clear, Save and Load configurations

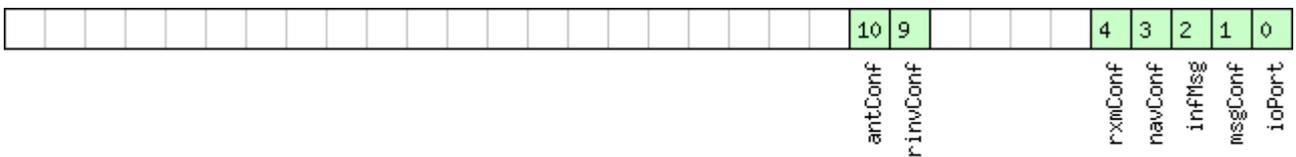
Message	CFG-CFG					
Description	Clear, Save and Load configurations					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Command					
Comment	See Receiver Configuration for a detailed description on how Receiver Configuration should be used. The three masks are made up of individual bits, each bit indicating the sub-section of all configurations on which the corresponding action shall be carried out. The reserved bits in the masks must be set to '0'. For detailed information refer to the Organization of the Configuration Sections . Note that commands can be combined. The sequence of execution is Clear, Save, Load					
Message Structure	Header	Class	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	

CFG-CFG continued

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

Bitfield clearMask

This Graphic explains the bits of clearMask

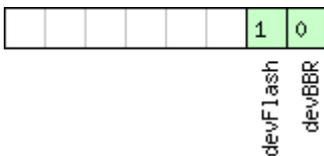


- signed value
- unsigned value
- reserved

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



- signed value
- unsigned value
- reserved

Name	Description
devBBR	device battery backed RAM
devFlash	device Flash

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

30.3.1 Poll Datum Setting

Message	CFG-DAT					
Description	Poll Datum Setting					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Upon sending of this message, the receiver returns CFG-DAT as defined below					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x06	0	see below	CK_A CK_B
No payload						

30.3.2 Set User-defined Datum

Message	CFG-DAT					
Description	Set User-defined Datum					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	-					
Message Structure	Header	Class	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	-	ma jA	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).	
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).	

30.3.3 The currently defined Datum

Message	CFG-DAT					
Description	The currently defined Datum					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Output					
Comment	Returns the parameters of the currently defined datum. If no user-defined datum has been set, this will default to WGS84.					
Message Structure	Header	Class	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: 0 = WGS84, -1 = user-defined	
2	CH[6]	-	datumName	-	ASCII String: WGS84 or USER	
8	R8	-	ma jA	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).	

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

30.4.1 Poll the GNSS system configuration

Message	CFG-GNSS					
Description	Poll the GNSS system configuration					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Polls the configuration of the GNSS system configuration					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x3E	0	see below	CK_A CK_B
No payload						

30.4.2 GNSS system configuration

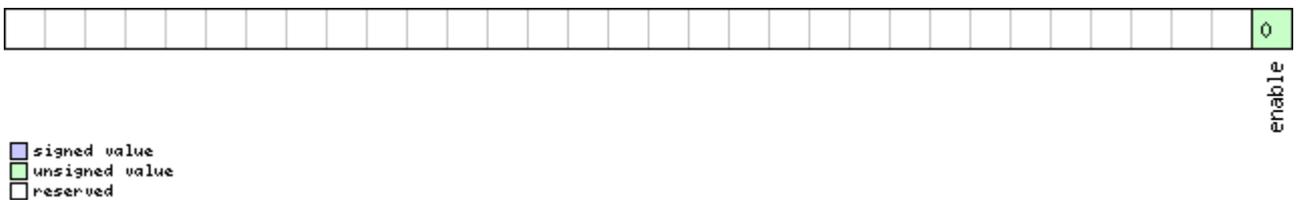
Message	CFG-GNSS					
Description	GNSS system configuration					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	<p>Gets or sets the GNSS system channel sharing configuration. The receiver will send an UBX-ACK-ACK message if the configuration is valid, an UBX-ACK-NAK if any configuration parameter is invalid.</p> <p>The number of tracking channels in use must not exceed the number of tracking channels available on hardware, and the sum of all reserved tracking channels needs to be smaller or equal the number of tracking channels in use. Additionally, the maximum number of tracking channels used for the specific GNSS system must be greater or equal to the number of reserved tracking channels.</p> <p>See section GNSS Configuration for a discussion of the use of this message and section Satellite Numbering for a description of the GNSS IDs available.</p> <p>Configuration specific to the GNSS system can be done via other messages. Configuration specific to SBAS can be done with CFG-SBAS.</p> <p>Note that GLONASS or BeiDou operation cannot be selected when the receiver is configured to operate in Power Save Mode (using CFG-RXM).</p> <p>GPS and QZSS should always be either both enabled or both disabled.</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x3E	4 + 8*numConfigBlocks	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	msgVer	-	Message version (=0 for this version)	
1	U1	-	numTrkChHw	-	Number of tracking channels available in hardware (read only)	
2	U1	-	numTrkChUse	-	Number of tracking channels to use (<= numTrkChHw)	
3	U1	-	numConfigBlocks	-	Number of configuration blocks following	
Start of repeated block (numConfigBlocks times)						

CFG-GNSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4 + 8*N	U1	-	gnssId	-	GNSS identifier (see Satellite Numbering)
5 + 8*N	U1	-	resTrkCh	-	Number of reserved (minimum) tracking channels for this GNSS system
6 + 8*N	U1	-	maxTrkCh	-	Maximum number of tracking channels used for this GNSS system (>=resTrkChn)
7 + 8*N	U1	-	reserved1	-	Reserved
8 + 8*N	X4	-	flags	-	bitfield of flags (see graphic below)
End of repeated block					

Bitfield flags

This Graphic explains the bits of flags



Name	Description
enable	Enable this GNSS system

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

30.5.1 Poll INF message configuration for one protocol

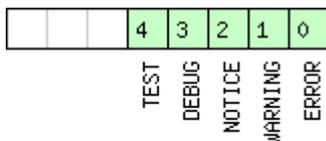
Message	CFG-INF					
Description	Poll INF message configuration for one protocol					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x02	1	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	protocolID	-	Protocol Identifier, identifying the output protocol for this Poll Request. The following are valid Protocol Identifiers: 0: UBX Protocol 1: NMEA Protocol 2-255: Reserved	

30.5.2 Information message configuration

Message	CFG-INF					
Description	Information message configuration					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	The value of infMsgMask[x] below are that each bit represents one of the INF class messages (Bit 0 for ERROR, Bit 1 for WARNING and so on.). For a complete list, see the Message Class INF . Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length. Output messages from the module contain only one configuration unit. Note that I/O Ports 1 and 2 correspond to serial ports 1 and 2. I/O port 0 is DDC. I/O port 3 is USB. I/O port 4 is SPI. I/O port 5 is reserved for future use.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x02	0 + 10*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
Start of repeated block (N times)						
N*10	U1	-	protocolID	-	Protocol Identifier, identifying for which protocol the configuration is set/get. The following are valid Protocol Identifiers: 0: UBX Protocol 1: NMEA Protocol 2-255: Reserved	
1 + 10*N	U1	-	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 port (see graphic below)	
End of repeated block						

Bitfield infMsgMask

This Graphic explains the bits of infMsgMask



- signed value
- unsigned value
- reserved

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

30.6 UBX-CFG-ITFM (0x06 0x39)

30.6.1 Poll Jamming/Interference Monitor configuration.

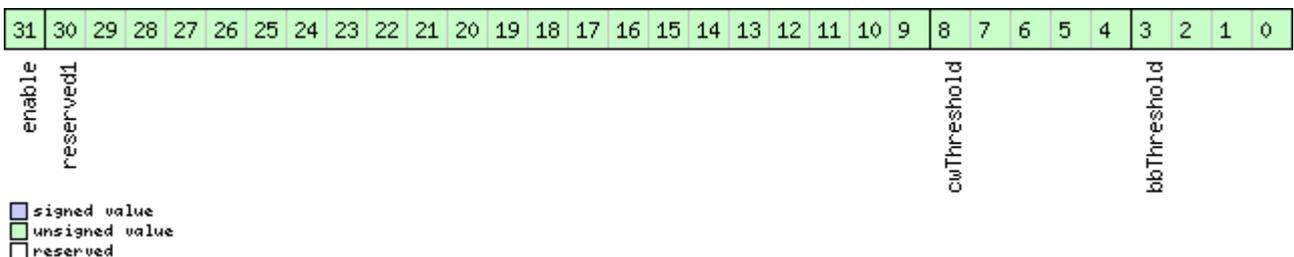
Message	CFG-ITFM					
Description	Poll Jamming/Interference Monitor configuration.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x39	0	see below	CK_A CK_B
No payload						

30.6.2 Jamming/Interference Monitor configuration.

Message	CFG-ITFM					
Description	Jamming/Interference Monitor configuration.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Command					
Comment	Configuration of Jamming/Interference monitor.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x39	8	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	X4	-	config	-	interference config word. (see graphic below)	
4	X4	-	config2	-	extra settings for jamming/interference monitor (see graphic below)	

Bitfield config

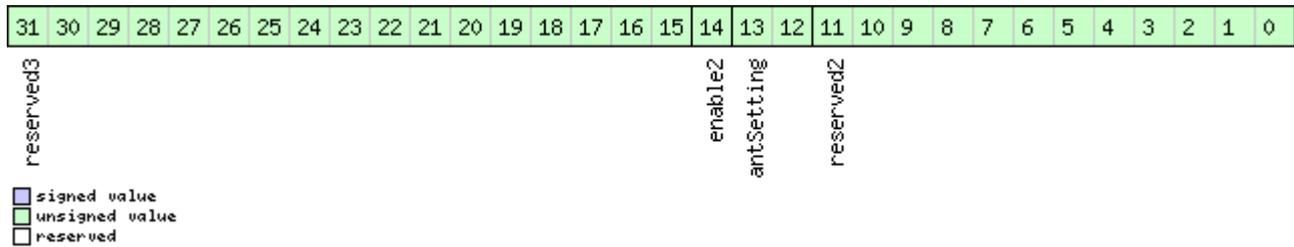
This Graphic explains the bits of config



Name	Description
bbThreshold	Broadband jamming detection threshold (unit = dB)
cwThreshold	CW jamming detection threshold (unit = dB)
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
enable2	Set to 1 to scan auxiliary bands (u-blox M8 only, otherwise ignored)
reserved3	reserved, set to 0

30.7 UBX-CFG-LOGFILTER (0x06 0x47)

30.7.1 Poll Data Logger filter Configuration

Message	CFG-LOGFILTER					
Description	Poll Data Logger filter Configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Upon sending of this message, the receiver returns CFG-LOGFILTER as defined below					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x47	0	see below	CK_A CK_B
No payload						

30.7.2 Data Logger Configuration

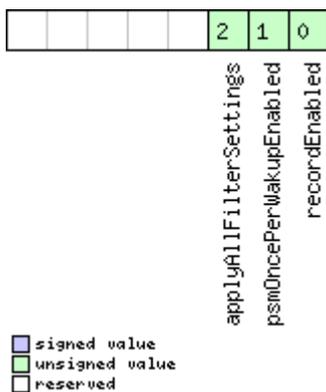
Message	CFG-LOGFILTER					
Description	Data Logger Configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	This message is used to enable/disable logging and to get or set the position entry filter settings. Position entries can be filtered based on time difference, position difference or current speed thresholds. Position and speed filtering also have a minimum time interval. A position is logged if any of the thresholds are exceeded. If a threshold is set to zero it is ignored. The maximum rate of position logging is 1Hz. The filter settings will only be applied if the 'applyAllFilterSettings' flag is set. This enables recording to be enabled/disabled without affecting the other settings.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x47	12	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	The version of this message. Set to 1	

CFG-LOGFILTER continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
1	X1	-	flags	-	Flags (see graphic below)
2	U2	-	minInterval	s	Minimum time interval between logged positions (0 = not set). This is only applied in combination with the speed and/or position thresholds
4	U2	-	timeThreshold	s	If the time difference is greater than the threshold then the position is logged (0 = not set).
6	U2	-	speedThreshold	m/s	If the current speed is greater than the threshold then the position is logged (0 = not set). minInterval also applies
8	U4	-	positionThreshold	m	If the 3D position difference is greater than the threshold then the position is logged (0 = not set). minInterval also applies

Bitfield flags

This Graphic explains the bits of flags



Name	Description
recordEnabled	1 = enable recording, 0 = disable recording
psmOncePerWakeupEnabled	1 = enable recording only one single position per PSM on/off mode wake up period, 0 = disable once per wake up
applyAllFilterSettings	1 = apply all filter settings, 0 = only apply recordEnabled

30.8 UBX-CFG-MSG (0x06 0x01)

30.8.1 Poll a message configuration

<i>Message</i>	CFG-MSG					
<i>Description</i>	Poll a message configuration					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	-					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x01	2	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	msgClass	-	Message Class	
1	U1	-	msgID	-	Message Identifier	

30.8.2 Set Message Rate(s)

<i>Message</i>	CFG-MSG					
<i>Description</i>	Set Message Rate(s)					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Input/Output					
<i>Comment</i>	Set/Get message rate configuration (s) to/from the receiver. See also section How to change between protocols . <ul style="list-style-type: none"> 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. 					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x01	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	msgClass	-	Message Class	
1	U1	-	msgID	-	Message Identifier	
2	U1[6]	-	rate	-	Send rate on I/O Port (6 Ports)	

30.8.3 Set Message Rate

Message	CFG-MSG					
Description	Set Message Rate					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	Set message rate configuration for the current port. See also section How to change between protocols .					
Message Structure	Header	Class	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 Port	

30.9 UBX-CFG-NAV5 (0x06 0x24)

30.9.1 Poll Navigation Engine Settings

Message	CFG-NAV5					
Description	Poll Navigation Engine Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x24	0	see below	CK_A CK_B
No payload						

30.9.2 Navigation Engine Settings

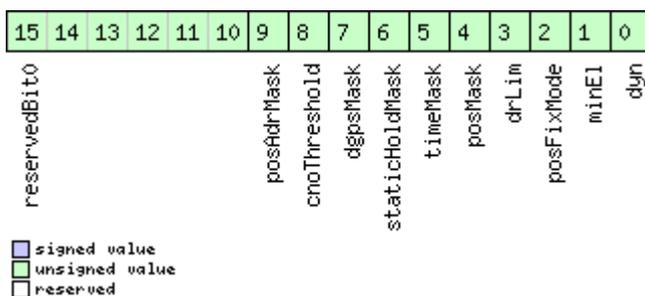
Message	CFG-NAV5					
Description	Navigation Engine Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	See the Navigation Configuration Settings Description for a detailed description of how these settings affect receiver operation.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	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)	

CFG-NAV5 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
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	Reserved
14	U2	0.1	pDop	-	Position DOP Mask to use
16	U2	0.1	tDop	-	Time DOP Mask to use
18	U2	-	pAcc	m	Position Accuracy Mask
20	U2	-	tAcc	m	Time Accuracy Mask
22	U1	-	staticHoldThreshold	cm/s	Static hold threshold
23	U1	-	dgpsTimeOut	s	DGPS timeout.
24	U1	-	cnoThreshNumSVs	-	Number of satellites required to have C/N0 above cnoThresh for a fix to be attempted
25	U1	-	cnoThresh	dBHz	C/N0 threshold for deciding whether to attempt a fix
26	U2	-	pAccAdr	m	ADR Position Accuracy Mask
28	U2	-	staticHoldMaxDist	m	Static hold distance threshold (before quitting static hold)
30	U2	-	reserved3	-	Always set to zero
32	U4	-	reserved4	-	Always set to zero

Bitfield mask

This Graphic explains the bits of mask



Bitfield mask Description continued

Name	Description
<i>Name</i>	<i>Description</i>
dyn	Apply dynamic model settings
minEl	Apply minimum elevation settings
posFixMode	Apply fix mode settings
drLim	Reserved
posMask	Apply position mask settings
timeMask	Apply time mask settings
staticHoldMask	Apply static hold settings
dgpsMask	Apply DGPS settings.
cnoThreshold	Apply CNO threshold settings (cnoThresh, cnoThreshNumSVs).
posAdrMask	Apply ADR position mask settings.
reservedBit0	reserved

30.10 UBX-CFG-NAVX5 (0x06 0x23)

30.10.1 Poll Navigation Engine Expert Settings

Message	CFG-NAVX5					
<i>Description</i>	Poll Navigation Engine Expert Settings					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	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.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x23	0	see below	CK_A CK_B
<i>No payload</i>						

30.10.2 Navigation Engine Expert Settings

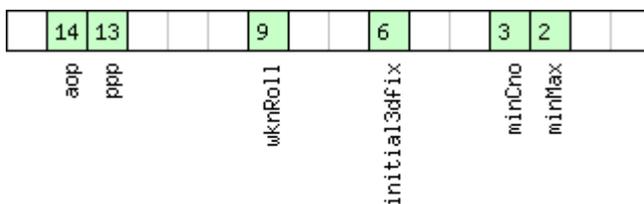
Message	CFG-NAVX5					
<i>Description</i>	Navigation Engine Expert Settings					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Input/Output					
<i>Comment</i>	-					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x23	40	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U2	-	version	-	Message version (0 for this version)	
2	X2	-	mask1	-	First Parameters Bitmask. Only the flagged parameters will be applied, unused bits must be set to 0. (see graphic below)	
4	U4	-	reserved0	-	Always set to zero	

CFG-NAVX5 continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
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	-	Only supported on certain product variants
27	U1	-	aopCfg	-	AssistNow Autonomous configuration (see graphic below)
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	U1	-	reserved14	-	Always set to zero
33	U1	-	reserved15	-	Always set to zero
34	U2	-	reserved3	-	Always set to zero
36	U4	-	reserved4	-	Always set to zero

Bitfield mask1

This Graphic explains the bits of mask1



- signed value
- unsigned value
- reserved

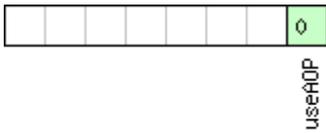
Name	Description
minMax	Apply min/max SVs settings
minCno	Apply minimum C/N0 setting
initial3dfix	Apply initial 3D fix settings
wknRoll	Apply GPS weeknumber rollover settings

Bitfield mask1 Description continued

Name	Description
ppp	Only supported on certain product variants
aop	Apply aopCfg (useAOP flag) and aopOrbMaxErr settings (AssistNow Autonomous)

Bitfield aopCfg

This Graphic explains the bits of aopCfg



signed value
 unsigned value
 reserved

Name	Description
useAOP	AOP enabled flag

30.11 UBX-CFG-NMEA (0x06 0x17)

30.11.1 Poll the NMEA protocol configuration

Message	CFG-NMEA					
Description	Poll the NMEA protocol configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x17	0	see below	CK_A CK_B
No payload						

30.11.2 NMEA protocol configuration (deprecated)

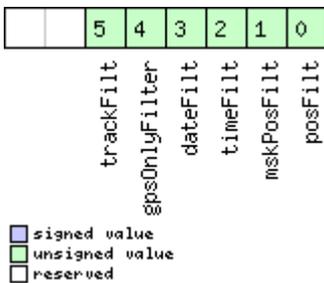
Message	CFG-NMEA					
Description	NMEA protocol configuration (deprecated)					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	<p>This message version is provided for backwards compatibility only. Use the last version listed below instead (its fields are backwards compatible with this version, it just has extra fields defined).</p> <p>Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.</p>					
Message Structure	Header	Class	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)	

CFG-NMEA continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
1	U1	-	nmeaVersion	-	0x23: NMEA version 2.3 0x21: NMEA version 2.1
2	U1	-	numSV	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs
3	X1	-	flags	-	flags (see graphic below)

Bitfield filter

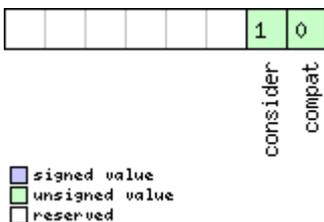
This Graphic explains the bits of filter



Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

This Graphic explains the bits of flags



Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.

30.11.3 NMEA protocol configuration V0 (deprecated)

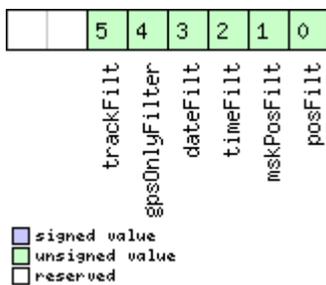
Message		CFG-NMEA					
Description		NMEA protocol configuration V0 (deprecated)					
Firmware		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type		Input/Output					
Comment		<p>This message version is provided for backwards compatibility only. Use the last version listed below instead (its fields are backwards compatible with this version, it just has extra fields defined).</p> <p>Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.</p>					
Message Structure		Header	Class	ID	Length (Bytes)	Payload	Checksum
		0xB5 0x62	0x06	0x17	12	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	X1	-	filter	-	filter flags (see graphic below)		
1	U1	-	nmeaVersion	-	0x23: NMEA version 2.3 0x21: NMEA version 2.1		
2	U1	-	numSV	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs		
3	X1	-	flags	-	flags (see graphic below)		
4	X4	-	gnssToFilter	-	Filters out satellites based on their GNSS. If a bitfield is enabled, the corresponding satellites will be not output. (see graphic below)		
8	U1	-	svNumbering	-	Configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID. 0: Strict - Satellites are not output 1: Extended - Use proprietary numbering (see Satellite numbering)		
9	U1	-	mainTalkerId	-	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see UBX-CFG-GNSS). This field enables the main Talker ID to be overridden. 0: Main Talker ID is not overridden 1: Set main Talker ID to 'GP' 2: Set main Talker ID to 'GL' 3: Set main Talker ID to 'GN' 4: Set main Talker ID to 'GA' 5: Set main Talker ID to 'GB'		

CFG-NMEA continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
10	U1	-	gsvTalkerId	-	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden. 0: Use GNSS specific Talker ID (as defined by NMEA) 1: Use the main Talker ID
11	U1	-	version	-	Message version (set to 0 for this version)

Bitfield filter

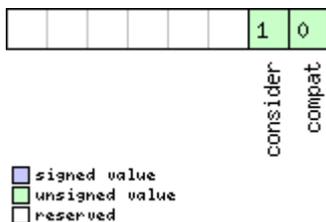
This Graphic explains the bits of filter



Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

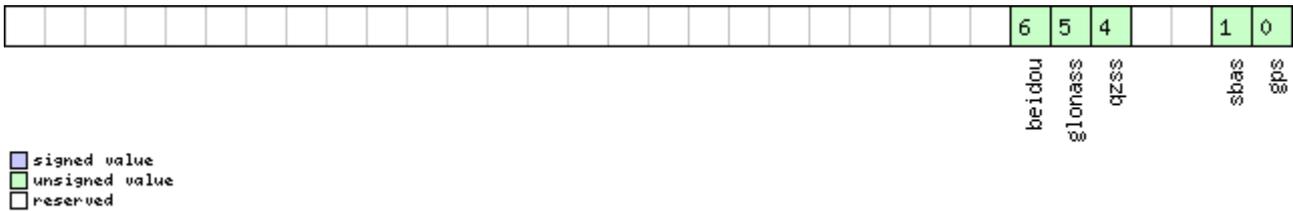
This Graphic explains the bits of flags



Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.

Bitfield gnsstoFilter

This Graphic explains the bits of gnsstoFilter



Name	Description
gps	Disable reporting of GPS satellites
sbas	Disable reporting of SBAS satellites
qzss	Disable reporting of QZSS satellites
glonass	Disable reporting of GLONASS satellites
beidou	Disable reporting of BeiDou satellites

30.11.4 Extended NMEA protocol configuration V1

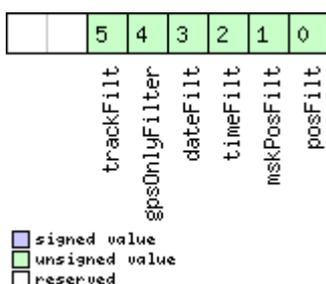
Message	CFG-NMEA					
Description	Extended NMEA protocol configuration V1					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	Set/Get the NMEA protocol configuration. See section NMEA Protocol Configuration for a detailed description of the configuration effects on NMEA output.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x17	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	X1	-	filter	-	filter flags (see graphic below)	
1	U1	-	nmeaVersion	-	0x41: NMEA version 4.1 0x40: NMEA version 4.0 0x23: NMEA version 2.3 0x21: NMEA version 2.1	
2	U1	-	numSV	-	Maximum Number of SVs to report per TalkerId. 0: unlimited 8: 8 SVs 12: 12 SVs 16: 16 SVs	
3	X1	-	flags	-	flags (see graphic below)	
4	X4	-	gnsstoFilter	-	Filters out satellites based on their GNSS. If a bitfield is enabled, the corresponding satellites will be not output. (see graphic below)	

CFG-NMEA continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
8	U1	-	svNumbering	-	Configures the display of satellites that do not have an NMEA-defined value. Note: this does not apply to satellites with an unknown ID. 0: Strict - Satellites are not output 1: Extended - Use proprietary numbering (see Satellite numbering)
9	U1	-	mainTalkerId	-	By default the main Talker ID (i.e. the Talker ID used for all messages other than GSV) is determined by the GNSS assignment of the receiver's channels (see UBX-CFG-GNSS). This field enables the main Talker ID to be overridden. 0: Main Talker ID is not overridden 1: Set main Talker ID to 'GP' 2: Set main Talker ID to 'GL' 3: Set main Talker ID to 'GN' 4: Set main Talker ID to 'GA' 5: Set main Talker ID to 'GB'
10	U1	-	gsvTalkerId	-	By default the Talker ID for GSV messages is GNSS specific (as defined by NMEA). This field enables the GSV Talker ID to be overridden. 0: Use GNSS specific Talker ID (as defined by NMEA) 1: Use the main Talker ID
11	U1	-	version	-	Message version (set to 1 for this version)
12	CH[2]	-	bdsTalkerId	-	Sets the two characters that should be used for the BeiDou Talker ID If these are set to zero, the default BeiDou TalkerId will be used
14	U2	-	reserved1	-	Reserved, always set to 0
16	U4	-	reserved2	-	Reserved, always set to 0

Bitfield filter

This Graphic explains the bits of filter



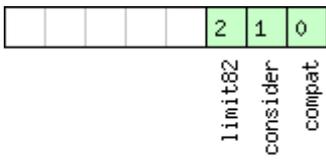
Name	Description
------	-------------

Bitfield filter Description continued

Name	Description
posFilt	Enable position output for failed or invalid fixes
mskPosFilt	Enable position output for invalid fixes
timeFilt	Enable time output for invalid times
dateFilt	Enable date output for invalid dates
gpsOnlyFilter	Restrict output to GPS satellites only
trackFilt	Enable COG output even if COG is frozen

Bitfield flags

This Graphic explains the bits of flags

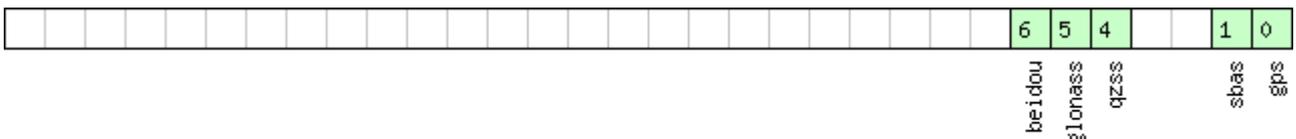


- signed value
- unsigned value
- reserved

Name	Description
compat	enable compatibility mode. This might be needed for certain applications when customer's NMEA parser expects a fixed number of digits in position coordinates
consider	enable considering mode.
limit82	enable strict limit to 82 characters maximum.

Bitfield gnsstoFilter

This Graphic explains the bits of gnsstoFilter



- signed value
- unsigned value
- reserved

Name	Description
gps	Disable reporting of GPS satellites
sbas	Disable reporting of SBAS satellites
qzss	Disable reporting of QZSS satellites
glonass	Disable reporting of GLONASS satellites
beidou	Disable reporting of BeiDou satellites

30.12 UBX-CFG-ODO (0x06 0x1E)

30.12.1 Poll Odometer, Low-speed COG Engine Settings

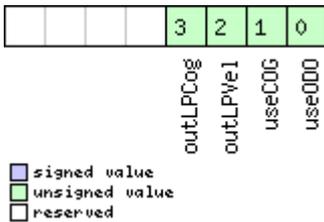
Message	CFG-ODO					
Description	Poll Odometer, Low-speed COG Engine Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a message of type CFG-ODO with a payload as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1E	0	see below	CK_A CK_B
No payload						

30.12.2 Odometer, Low-speed COG Engine Settings

Message	CFG-ODO					
Description	Odometer, Low-speed COG Engine Settings					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1E	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	Message version (0 for this version)	
1	U1	-	reserved0	-	Reserved	
2	U2	-	reserved1	-	Reserved	
4	U1	-	flags	-	Odometer/Low-speed COG filter flags (see graphic below)	
5	X1	-	odoCfg	-	Odometer filter settings (see graphic below)	
6	U2	-	reserved3	-	Reserved	
8	U4	-	reserved4	-	Reserved	
12	U4	-	reserved5	-	Reserved	
16	U1	-	velLpGain	-	Velocity low-pass filter level, range 0..255	
17	U1	-	cogLpGain	-	COG low-pass filter level (at speed < 8 m/s), range 0..255	
18	U2	-	reserved6	-	Reserved	

Bitfield flags

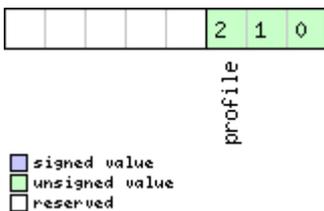
This Graphic explains the bits of flags



Name	Description
useODO	Odometer enabled flag
useCOG	Low-speed COG filter enabled flag
outLPVcl	Output low-pass filtered velocity flag
outLPCog	Output low-pass filtered heading (COG) flag

Bitfield odoCfg

This Graphic explains the bits of odoCfg



Name	Description
profile	Profile type (0=running, 1=cycling, 2=swimming, 3=car, 4=custom)

30.13 UBX-CFG-PM2 (0x06 0x3B)

30.13.1 Poll extended Power Management configuration

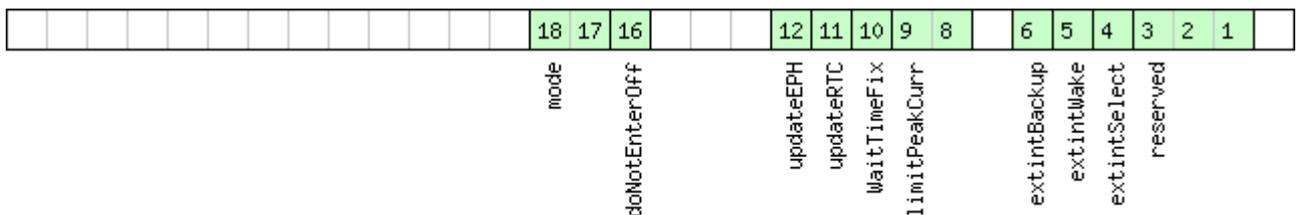
Message	CFG-PM2					
Description	Poll extended Power Management configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x3B	0	see below	CK_A CK_B
No payload						

30.13.2 Extended Power Management configuration

Message		CFG-PM2					
Description		Extended Power Management configuration					
Firmware		Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type		Input/Output					
Comment		-					
Message Structure		Header	Class	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 (1 for this version)		
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		
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



- signed value
- unsigned value
- reserved

Name	Description
reserved	Reserved: Must be set to '000'

Bitfield flags Description continued

Name	Description
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 <i>inactive for search</i> state 1 receiver does not enter <i>inactive for search</i> 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

30.14 UBX-CFG-PRT (0x06 0x00)

30.14.1 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: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Polls the configuration of the I/O Port on which this message is received					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	0	see below	CK_A CK_B
No payload						

30.14.2 Polls the configuration for one I/O Port

<i>Message</i>		CFG-PRT					
<i>Description</i>		Polls the configuration for one I/O Port					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>		Poll Request					
<i>Comment</i>		Sending this message with a port ID as payload results in having the receiver return the configuration for the specified port.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x06	0x00	1	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		
0	U1	-	PortID	-	Port Identifier Number (see the other versions of CFG-PRT for valid values)		

30.14.3 Port Configuration for UART

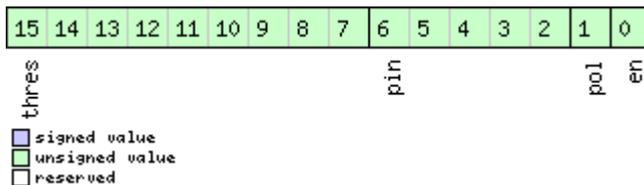
<i>Message</i>		CFG-PRT					
<i>Description</i>		Port Configuration for UART					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>		Input/Output					
<i>Comment</i>		Several configurations can be concatenated to one input message. In this case the payload length can be a multiple of the normal length (see the other versions of CFG-PRT). Output messages from the module contain only one configuration unit. Note that this message can affect baud rate and other transmission parameters. Because there may be messages queued for transmission there may be uncertainty about which protocol applies to such messages. In addition a message currently in transmission may be corrupted by a protocol change. Host data reception parameters may have to be changed to be able to receive future messages, including the acknowledge message resulting from the CFG-PRT message.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		
0	U1	-	portID	-	Port Identifier Number (see Serial Communication Ports Description for valid UART port IDs)		
1	U1	-	reserved0	-	Reserved		
2	X2	-	txReady	-	TX ready PIN configuration (see graphic below)		
4	X4	-	mode	-	A bit mask describing the UART mode (see graphic below)		
8	U4	-	baudRate	Bits/s	Baud rate in bits/second		

CFG-PRT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
12	X2	-	inProtoMask	-	A mask describing which input protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
14	X2	-	outProtoMask	-	A mask describing which output protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
16	X2	-	flags	-	Flags bit mask (see graphic below)
18	U2	-	reserved5	-	Always set to zero

Bitfield txReady

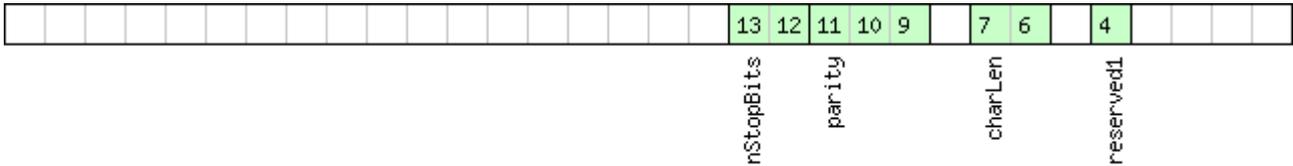
This Graphic explains the bits of txReady



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 $\geq \text{thres} * 8$ 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

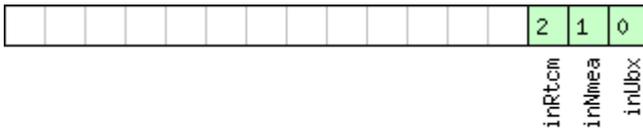


- signed value
- unsigned value
- reserved

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

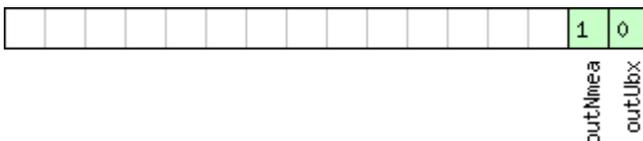


- signed value
- unsigned value
- reserved

Name	Description
inUbx	UBX protocol
inNmea	NMEA protocol
inRtcm	RTCM protocol

Bitfield outProtoMask

This Graphic explains the bits of outProtoMask



- signed value
- unsigned value
- reserved

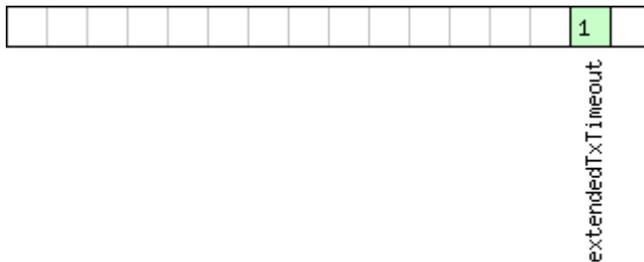
Name	Description
outUbx	UBX protocol

Bitfield outProtoMask Description continued

Name	Description
outNmea	NMEA protocol

Bitfield flags

This Graphic explains the bits of flags



- signed value
- unsigned value
- reserved

Name	Description
extendedTxTimeout	Extended TX timeout: if set, the port will timeout if allocated TX memory ≥ 4 kB and no activity for 1.5s.

30.14.4 Port Configuration for USB Port

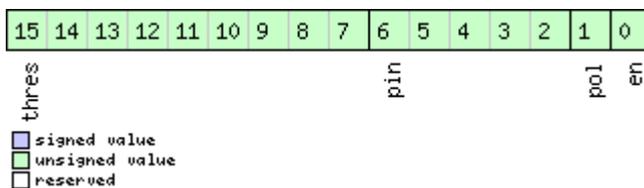
Message	CFG-PRT					
Description	Port Configuration for USB Port					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be 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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	portID	-	Port Identifier Number (= 3 for USB port)	
1	U1	-	reserved0	-	Reserved	
2	X2	-	txReady	-	TX ready PIN configuration (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)	

CFG-PRT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
14	X2	-	outProtoMask	-	A mask describing which output protocols are active. Each bit of this mask is used for a protocol. Through that, multiple protocols can be defined on a single port. (see graphic below)
16	U2	-	reserved4	-	Always set to zero
18	U2	-	reserved5	-	Always set to zero

Bitfield txReady

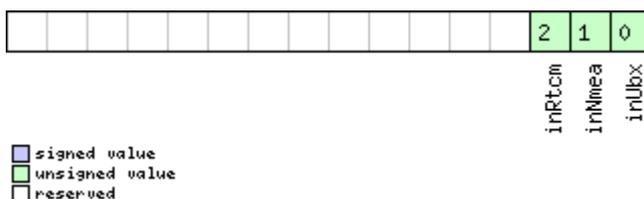
This Graphic explains the bits of txReady



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 $\geq \text{thres} * 8$ 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

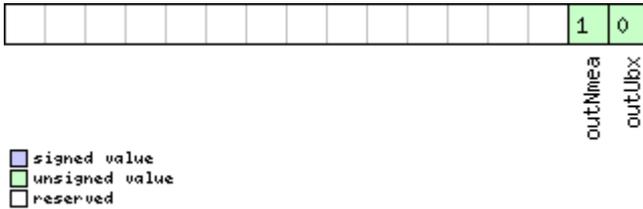
This Graphic explains the bits of inProtoMask



Name	Description
inUbx	UBX protocol
inNmea	NMEA protocol
inRtcm	RTCM protocol

Bitfield outProtoMask

This Graphic explains the bits of outProtoMask



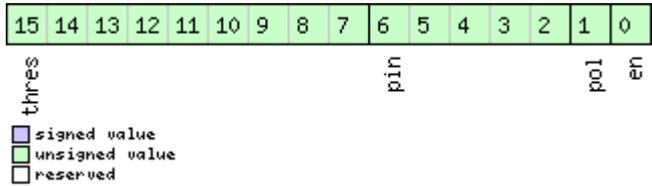
Name	Description
outUbx	UBX protocol
outNmea	NMEA protocol

30.14.5 Port Configuration for SPI Port

Message	CFG-PRT					
Description	Port Configuration for SPI Port					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	portID	-	Port Identifier Number (= 4 for SPI port)	
1	U1	-	reserved0	-	Reserved	
2	X2	-	txReady	-	TX ready PIN configuration (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	X2	-	flags	-	Flags bit mask (see graphic below)	
18	U2	-	reserved5	-	Always set to zero	

Bitfield txReady

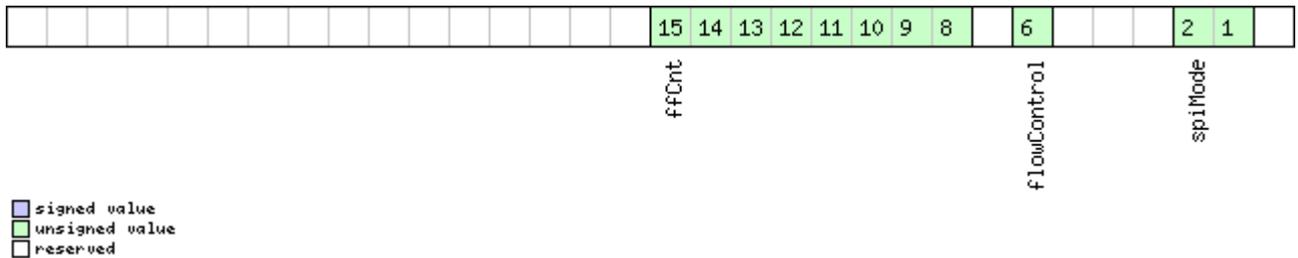
This Graphic explains the bits of txReady



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 \geq 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

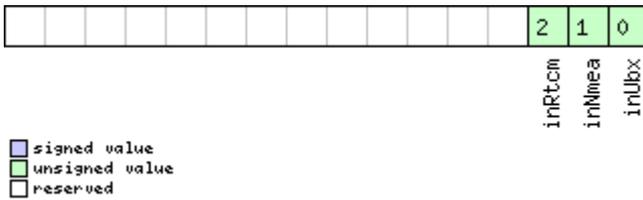
This Graphic explains the bits of mode



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

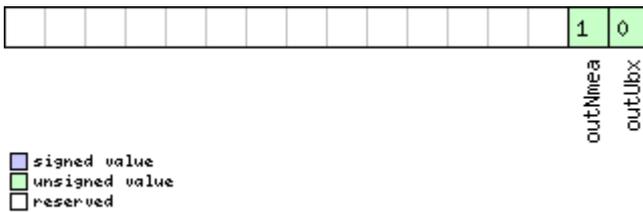
Bitfield inProtoMask

This Graphic explains the bits of inProtoMask



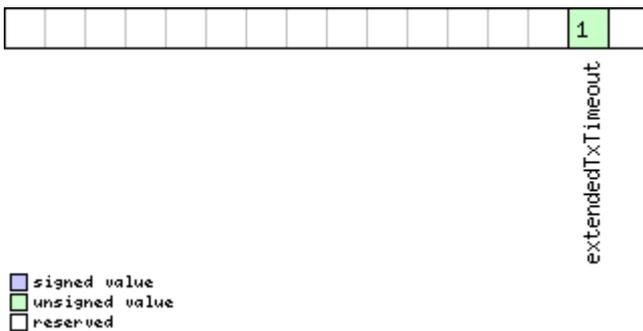
Bitfield outProtoMask

This Graphic explains the bits of outProtoMask



Bitfield flags

This Graphic explains the bits of flags



Name	Description
extendedTxTimeout	Extended TX timeout: if set, the port will timeout if allocated TX memory >=4 kB and no activity for 1.5s.

30.14.6 Port Configuration for DDC Port

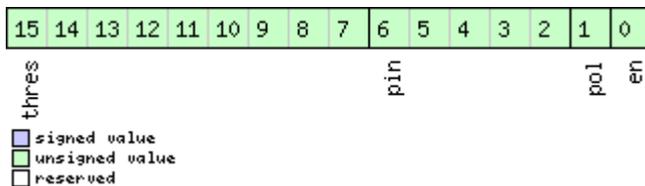
Message	CFG-PRT					
Description	Port Configuration for DDC Port					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	Several configurations can be concatenated to one input message. In this case the payload length can be 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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x00	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	

CFG-PRT continued

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	-	TX ready PIN configuration (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	X2	-	flags	-	Flags bit mask (see graphic below)
18	U2	-	reserved5	-	Always set to zero

Bitfield txReady

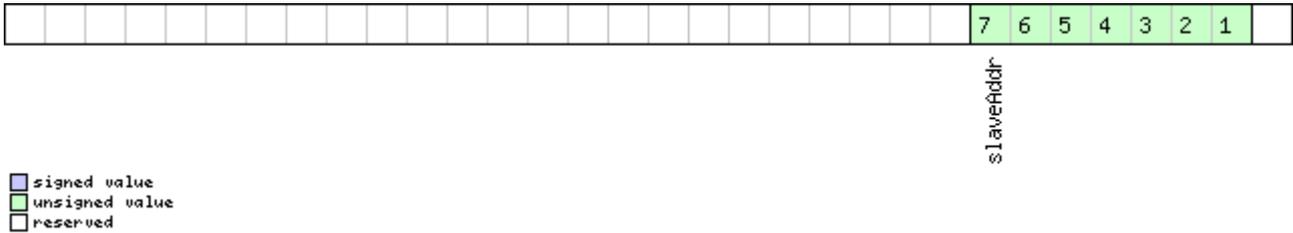
This Graphic explains the bits of txReady



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 \geq 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

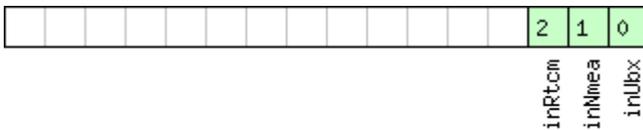


- signed value
- unsigned value
- reserved

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

Bitfield inProtoMask

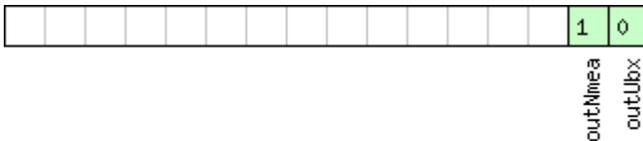
This Graphic explains the bits of inProtoMask



- signed value
- unsigned value
- reserved

Bitfield outProtoMask

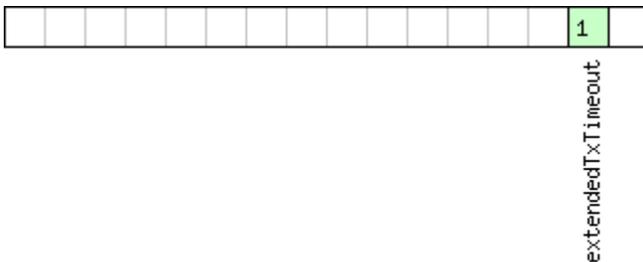
This Graphic explains the bits of outProtoMask



- signed value
- unsigned value
- reserved

Bitfield flags

This Graphic explains the bits of flags



- signed value
- unsigned value
- reserved

Name	Description
extendedTxTimeout	Extended TX timeout: if set, the port will timeout if allocated TX memory >=4 kB and no activity for 1.5s.

30.15 UBX-CFG-PWR (0x06 0x57)

30.15.1 Put the receiver in a defined power state

<i>Message</i>	CFG-PWR					
<i>Description</i>	Put the receiver in a defined power state					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Set					
<i>Comment</i>	-					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x57	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	version	-	Message version (1 for this version)	
1	U1	-	reserved1	-	reserved	
2	U2	-	reserved2	-	reserved	
4	U4	-	state	-	Enter system state 0x52554E20: GNSS running 0x53544F50: GNSS stopped 0x42434B50: Software Backup	

30.16 UBX-CFG-RATE (0x06 0x08)

30.16.1 Poll Navigation/Measurement Rate Settings

<i>Message</i>	CFG-RATE					
<i>Description</i>	Poll Navigation/Measurement Rate Settings					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	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					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x08	0	see below	CK_A CK_B
<i>No payload</i>						

30.16.2 Navigation/Measurement Rate Settings

<i>Message</i>	CFG-RATE					
<i>Description</i>	Navigation/Measurement Rate Settings					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Input/Output					
<i>Comment</i>	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. <ul style="list-style-type: none"> The update rate has a direct influence on the power consumption. The more fixes that are required, the more CPU power and communication resources are required. For most applications a 1 Hz update rate would be sufficient. When using Power Save Mode, measurement and navigation rate can differ from the values configured here. See Measurement and navigation rate with Power Save Mode for details. 					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x08	6	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U2	-	measRate	ms	Measurement Rate, GPS measurements are taken every measRate milliseconds	
2	U2	-	navRate	cycles	Navigation Rate, in number of measurement cycles. This parameter cannot be changed, and must be set to 1.	
4	U2	-	timeRef	-	Alignment to reference time: 0 = UTC time, 1 = GPS time	

30.17 UBX-CFG-RINV (0x06 0x34)

30.17.1 Poll contents of Remote Inventory

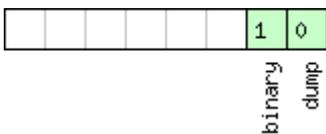
<i>Message</i>	CFG-RINV					
<i>Description</i>	Poll contents of Remote Inventory					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	-					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x34	0	see below	CK_A CK_B
<i>No payload</i>						

30.17.2 Contents of Remote Inventory

Message	CFG-RINV					
Description	Contents of Remote Inventory					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	If N is greater than 30, the excess bytes are discarded. In future firmware versions, this limit may change.					
Message Structure	Header	Class	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



signed value
 unsigned value
 reserved

Name	Description
dump	Dump data at startup. Does not work if flag binary is set.
binary	Data is binary

30.18 UBX-CFG-RST (0x06 0x04)

30.18.1 Reset Receiver / Clear Backup Data Structures

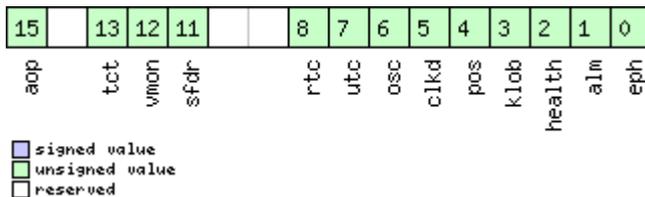
Message	CFG-RST					
Description	Reset Receiver / Clear Backup Data Structures					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Command					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x04	4	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	

CFG-RST continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	X2	-	navBbrMask	-	BBR Sections to clear. The following Special Sets apply: 0x0000 Hot start 0x0001 Warm start 0xFFFF Cold start (see graphic below)
2	U1	-	resetMode	-	Reset Type 0x00 - Hardware reset (Watchdog) immediately 0x01 - Controlled Software reset 0x02 - Controlled Software reset (GNSS only) 0x04 - Hardware reset (Watchdog) after shutdown 0x08 - Controlled GNSS stop 0x09 - Controlled GNSS start
3	U1	-	reserved1	-	Reserved

Bitfield navBbrMask

This Graphic explains the bits of navBbrMask



Name	Description
eph	Ephemeris
alm	Almanac
health	Health
klob	Klobuchar parameters
pos	Position
clkd	Clock Drift
osc	Oscillator Parameter
utc	UTC Correction + GPS Leap Seconds Parameters
rtc	RTC
sfdr	SFDR Parameters
vmon	SFDR Vehicle Monitoring Parameters
tct	TCT Parameters
aop	Autonomous Orbit Parameters

30.19 UBX-CFG-RXM (0x06 0x11)

30.19.1 Poll RXM configuration

<i>Message</i>	CFG-RXM					
<i>Description</i>	Poll RXM configuration					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	Upon sending of this message, the receiver returns CFG-RXM as defined below					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x11	0	see below	CK_A CK_B
No payload						

30.20 UBX-CFG-SBAS (0x06 0x16)

30.20.1 Poll contents of SBAS Configuration

<i>Message</i>	CFG-SBAS					
<i>Description</i>	Poll contents of SBAS Configuration					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Poll Request					
<i>Comment</i>	-					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x16	0	see below	CK_A CK_B
No payload						

30.20.2 SBAS Configuration

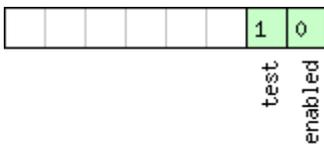
<i>Message</i>	CFG-SBAS					
<i>Description</i>	SBAS Configuration					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Input/Output					
<i>Comment</i>	This message configures the SBAS receiver subsystem (i.e. WAAS, EGNOS, MSAS). See the SBAS Configuration Settings Description for a detailed description of how these settings affect receiver operation.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x06	0x16	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
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 (obsolete and superseded by UBX-CFG-GNSS in protocol versions 14+).	

CFG-SBAS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
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

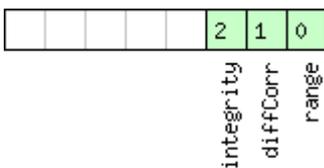


- signed value
- unsigned value
- reserved

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

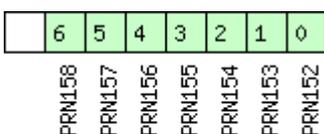


- signed value
- unsigned value
- reserved

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

Bitfield scanmode2

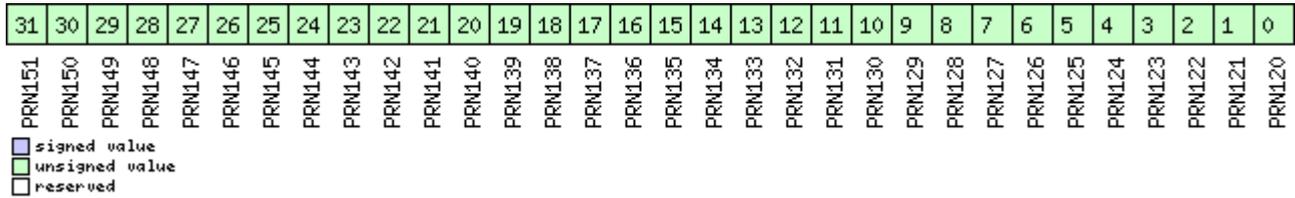
This Graphic explains the bits of scanmode2



- signed value
- unsigned value
- reserved

Bitfield scanmode1

This Graphic explains the bits of scanmode1



30.21 UBX-CFG-TP5 (0x06 0x31)

30.21.1 Poll Time Pulse Parameters

Message	CFG-TP5					
Description	Poll Time Pulse Parameters					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x31	0	see below	CK_A CK_B
No payload						

30.21.2 Poll Time Pulse Parameters

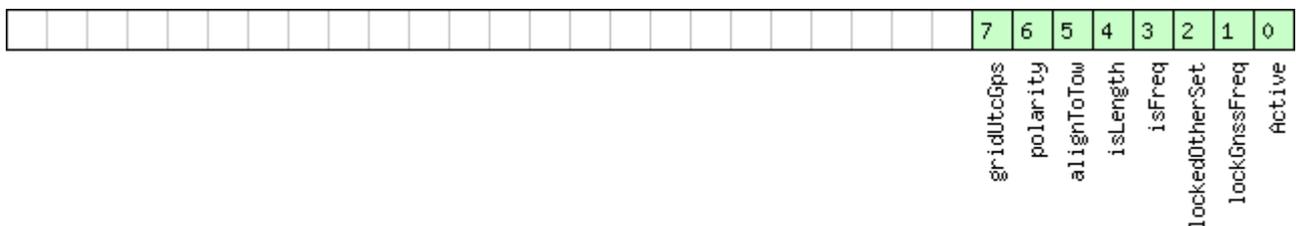
Message	CFG-TP5					
Description	Poll Time Pulse Parameters					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Sending this message to the receiver results in the receiver returning a message of type CFG-TP5 with a payload as defined below for the specified time pulse.					
Message Structure	Header	Class	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	-	Time pulse selection (0 = TIMEPULSE, 1 = TIMEPULSE2)	

30.21.3 Time Pulse Parameters

Message	CFG-TP5					
Description	Time Pulse Parameters					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	This message is used to get/set time pulse parameters. For more information see section Time pulse .					
Message Structure	Header	Class	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	-	Time pulse selection (0 = TIMEPULSE, 1 = TIMEPULSE2)	
1	U1	-	version	-	Version, 1 for this message	
2	U2	-	reserved1	-	Reserved	
4	I2	-	antCableDelay	ns	Antenna cable delay	
6	I2	-	rfGroupDelay	ns	RF group delay	
8	U4	-	freqPeriod	Hz_or_us	Frequency or period time, depending on setting of bit 'isFreq'	
12	U4	-	freqPeriodLock	Hz_or_us	Frequency or period time when locked to GNSS time, only used if 'lockedOtherSet' is set	
16	U4	-	pulseLenRatio	us_or_2^-32	Pulse length or duty cycle, depending on 'isLength'	
20	U4	-	pulseLenRatioLock	us_or_2^-32	Pulse length or duty cycle when locked to GNSS time, only used if 'lockedOtherSet' is set	
24	I4	-	userConfigDelay	ns	User configurable time pulse delay	
28	X4	-	flags	-	Configuration flags (see graphic below)	

Bitfield flags

This Graphic explains the bits of flags



- signed value
- unsigned value
- reserved

Name	Description
Active	if set enable time pulse; if pin assigned to another function, other function takes precedence
lockGnssFreq	if set synchronize time pulse to GNSS as soon as GNSS time is valid, otherwise use local clock

Bitfield flags Description continued

Name	Description
lockedOtherSet	If set use 'freqPeriodLock' and 'pulseLenRatioLock' as soon as GNSS time is valid and 'freqPeriod' and 'pulseLenRatio' if GNSS time is invalid; if flag is cleared 'freqPeriod' and 'pulseLenRatio' are used regardless of GNSS time.
isFreq	If set 'freqPeriodLock' and 'freqPeriod' are interpreted as frequency, otherwise interpreted as period.
isLength	If set 'pulseLenRatioLock' and 'pulseLenRatio' interpreted as pulse length, otherwise interpreted as duty cycle.
alignToTow	Align pulse to top of second (period time must be integer fraction of 1s). Also set 'lockGnssFreq' to use this feature.
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 This flag is only relevant if 'lockGnssFreq' and 'alignToTow' are set. Note that GPS time is estimated by the receiver if locked to any GNSS system. If the receiver has a valid GNSS fix it will attempt to steer the TP to the GPS time grid even if the GPS time is not based on information from GPS satellites. If you want to use purely GPS based timing you should also consider restricting the used GNSS constellations in CFG-GNSS .

30.22 UBX-CFG-USB (0x06 0x1B)

30.22.1 Poll a USB configuration

Message	CFG-USB					
Description	Poll a USB configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1B	0	see below	CK_A CK_B
No payload						

30.22.2 USB Configuration

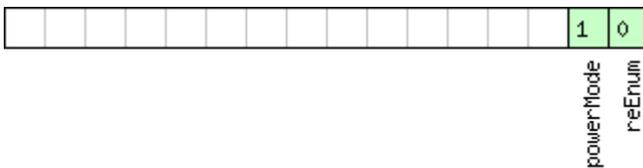
Message	CFG-USB					
Description	USB Configuration					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input/Output					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x06	0x1B	108	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	

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	mA	Power consumed by the device
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



- signed value
- unsigned value
- reserved

Name	Description
reEnum	force re-enumeration
powerMode	self-powered (1), bus-powered (0)

31 UBX-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.

31.1 UBX-INF-DEBUG (0x04 0x04)

31.1.1 ASCII String output, indicating debug output

Message	INF-DEBUG					
Description	ASCII String output, indicating debug output					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x04	0 + 1*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
Start of repeated block (N times)						
N*1	CH	-	str	-	ASCII Character	
End of repeated block						

31.2 UBX-INF-ERROR (0x04 0x00)

31.2.1 ASCII String output, indicating an error

Message	INF-ERROR					
Description	ASCII String output, indicating an error					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x00	0 + 1*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
Start of repeated block (N times)						
N*1	CH	-	str	-	ASCII Character	
End of repeated block						

31.3 UBX-INF-NOTICE (0x04 0x02)

31.3.1 ASCII String output, with informational contents

Message	INF-NOTICE					
Description	ASCII String output, with informational contents					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x02	0 + 1*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
Start of repeated block (N times)						
N*1	CH	-	str	-	ASCII Character	
End of repeated block						

31.4 UBX-INF-TEST (0x04 0x03)

31.4.1 ASCII String output, indicating test output

Message	INF-TEST					
Description	ASCII String output, indicating test output					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message has a variable length payload, representing an ASCII string.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x04	0x03	0 + 1*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
Start of repeated block (N times)						
N*1	CH	-	str	-	ASCII Character	
End of repeated block						

31.5 UBX-INF-WARNING (0x04 0x01)

31.5.1 ASCII String output, indicating a warning

<i>Message</i>	INF-WARNING					
<i>Description</i>	ASCII String output, indicating a warning					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Output					
<i>Comment</i>	This message has a variable length payload, representing an ASCII string.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x04	0x01	0 + 1*N	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
<i>Start of repeated block (N times)</i>						
N*1	CH	-	str	-	ASCII Character	
<i>End of repeated block</i>						

32 UBX-LOG (0x21)

Logging Messages: i.e. Log creation, deletion, info and retrieval.

The logging feature allows position fixes and arbitrary byte strings to be logged in flash memory attached to the receiver. For a full description of this feature see [Logging](#).

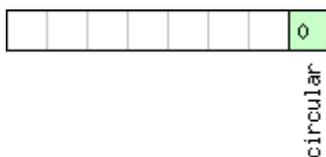
32.1 UBX-LOG-CREATE (0x21 0x07)

32.1.1 Create Log File

<i>Message</i>	LOG-CREATE					
<i>Description</i>	Create Log File					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Command					
<i>Comment</i>	This message is used to create an initial logging file and activate the logging subsystem. UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure. This message does not handle activation of recording or filtering of log entries (see UBX-CFG-LOGFILTER).					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x21	0x07	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	version	-	The version of this message. Set to 0	
1	X1	-	logCfg	-	Config flags (see graphic below)	
2	U1	-	reserved	-	Reserved. Set to zero	
3	U1	-	logSize	-	Indicates the size of the log: 0 (maximum safe size): Ensures that logging will not be interrupted and enough space will be left available for all other uses of the filestore 1 (minimum size): 2 (user defined): See 'userDefinedSize' below	
4	U4	-	userDefinedSize	bytes	Sets the maximum amount of space in the filestore that can be used by the logging task. This field is only applicable if logSize is set to user defined.	

Bitfield logCfg

This Graphic explains the bits of logCfg



- signed value
- unsigned value
- reserved

<i>Name</i>	<i>Description</i>
circular	Log is circular (new entries overwrite old ones in a full log) if this bit set

32.2 UBX-LOG-ERASE (0x21 0x03)

32.2.1 Erase Logged Data

Message	LOG-ERASE					
Description	Erase Logged Data					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Command					
Comment	This message deactivates the logging system and erases all logged data. UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x03	0	see below	CK_A CK_B
No payload						

32.3 UBX-LOG-FINDTIME (0x21 0x0E)

32.3.1 Find index of the first log entry <= given time

Message	LOG-FINDTIME					
Description	Find index of the first log entry <= given time					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message can be used to search a log for the index of the first entry less than or equal to the given time. This index can then be used with the UBX-LOG-RETRIEVE message to provide time-based retrieval of log entries.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0E	12	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	Message version (=0 for this version)	
1	U1	-	type	-	Message type, 0 for request	
2	U2	-	reserved1	-	Reserved	
4	U2	-	year	-	Year (1-65635) of UTC time	
6	U1	-	month	-	Month (1-12) of UTC time	
7	U1	-	day	-	Day (1-31) of UTC time	
8	U1	-	hour	-	Hour (0-23) of UTC time	
9	U1	-	minute	-	Minute (0-59) of UTC time	
10	U1	-	second	-	Second (0-60) of UTC time	
11	U1	-	reserved2	-	Reserved	

32.3.2 Response to FINDTIME request.

Message	LOG-FINDTIME					
Description	Response to FINDTIME request.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0E	8	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	Message version (=1 for this version)	
1	U1	-	type	-	Message type, 1 for response	
2	U2	-	reserved1	-	Reserved	
4	U4	-	entryNumber	-	Index of the most recent entry with time <= specified	

32.4 UBX-LOG-INFO (0x21 0x08)

32.4.1 Poll for log information

Message	LOG-INFO					
Description	Poll for log information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Upon sending of this message, the receiver returns UBX-LOG-INFO as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x08	0	see below	CK_A CK_B
No payload						

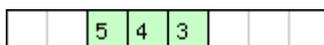
32.4.2 Log information

Message	LOG-INFO					
Description	Log information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message is used to report information about the logging subsystem. Note: <ul style="list-style-type: none"> The reported maximum log size will be smaller than that originally specified in LOG-CREATE due to logging and filestore implementation overheads. Log entries are compressed in a variable length fashion, so it may be difficult to predict log space usage with any precision. There may be times when the receiver does not have an accurate time (e.g. if the week number is not yet known), in which case some entries will not have a timestamp - this may result in the oldest/newest entry time values not taking account of these entries. 					

Message Structure		Header	Class	ID	Length (Bytes)	Payload	Checksum
		0xB5 0x62	0x21	0x08	48	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	U1	-	version	-	The version of this message. Set to 1		
1	U1[3]	-	reserved1	-	Reserved		
4	U4	-	filestoreCapacity	bytes	The capacity of the filestore		
8	U4	-	reserved2	-	Reserved		
12	U4	-	reserved3	-	Reserved		
16	U4	-	currentMaxLogSize	bytes	The maximum size the current log is allowed to grow to		
20	U4	-	currentLogSize	bytes	Approximate amount of space in log currently occupied		
24	U4	-	entryCount	-	Number of entries in the log. Note: for circular logs this value will decrease when a group of entries is deleted to make space for new ones.		
28	U2	-	oldestYear	-	Oldest entry UTC year year (1-65635) or zero if there are no entries with known time		
30	U1	-	oldestMonth	-	Oldest month (1-12)		
31	U1	-	oldestDay	-	Oldest day (1-31)		
32	U1	-	oldestHour	-	Oldest hour (0-23)		
33	U1	-	oldestMinute	-	Oldest minute (0-59)		
34	U1	-	oldestSecond	-	Oldest second (0-60)		
35	U1	-	reserved4	-	Reserved.		
36	U2	-	newestYear	-	Newest year (1-65635) or zero if there are no entries with known time		
38	U1	-	newestMonth	-	Newest month (1-12)		
39	U1	-	newestDay	-	Newest day (1-31)		
40	U1	-	newestHour	-	Newest hour (0-23)		
41	U1	-	newestMinute	-	Newest minute (0-59)		
42	U1	-	newestSecond	-	Newest second (0-60)		
43	U1	-	reserved5	-	Reserved.		
44	X1	-	status	-	Log status flags (see graphic below)		
45	U1[3]	-	reserved6	-	Reserved		

Bitfield status

This Graphic explains the bits of status



signed value
 unsigned value
 reserved

Name	Description
------	-------------

Bitfield status Description continued

Name	Description
recording	Log entry recording is currently turned on
inactive	Logging system not active - no log present
circular	The current log is circular

32.5 UBX-LOG-RETRIEVEPOSEXTRA (0x21 0x0f)

32.5.1 Odometer log entry

Message	LOG-RETRIEVEPOSEXTRA					
Description	Odometer log entry					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message is used to report an odometer log entry					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0f	32	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	entryIndex	-	The index of this log entry	
4	U1	-	version	-	The version of this message. Set to 0	
5	U1	-	reserved1	-	Reserved	
6	U2	-	year	-	Year (1-65635) of UTC time. Will be zero if time not known	
8	U1	-	month	-	Month (1-12) of UTC time	
9	U1	-	day	-	Day (1-31) of UTC time	
10	U1	-	hour	-	Hour (0-23) of UTC time	
11	U1	-	minute	-	Minute (0-59) of UTC time	
12	U1	-	second	-	Second (0-60) of UTC time	
13	U1	-	reserved2	-	Reserved	
14	U2	-	reserved3	-	Reserved	
16	U4	-	distance	-	Odometer distance travelled	
20	U4	-	reserved4	-	Reserved	
24	U4	-	reserved5	-	Reserved	
28	U4	-	reserved6	-	Reserved	

32.6 UBX-LOG-RETRIEVEPOS (0x21 0x0b)

32.6.1 Position fix log entry

Message	LOG-RETRIEVEPOS					
Description	Position fix log entry					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message is used to report a position fix log entry					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0b	40	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	entryIndex	-	The index of this log entry	
4	I4	1e-7	lon	deg	Longitude	
8	I4	1e-7	lat	deg	Latitude	
12	I4	-	hMSL	mm	Height above mean sea level	
16	U4	-	hAcc	mm	Horizontal accuracy estimate	
20	U4	-	gSpeed	mm/s	Ground speed (2-D)	
24	U4	-	heading	deg	Heading	
28	U1	-	version	-	The version of this message. Set to 0	
29	U1	-	fixType	-	Fix type: 2: 2D-Fix 3: 3D-Fix	
30	U2	-	year	-	Year (1-65635) of UTC time	
32	U1	-	month	-	Month (1-12) of UTC time	
33	U1	-	day	-	Day (1-31) of UTC time	
34	U1	-	hour	-	Hour (0-23) of UTC time	
35	U1	-	minute	-	Minute (0-59) of UTC time	
36	U1	-	second	-	Second (0-60) of UTC time	
37	U1	-	reserved1	-	Reserved	
38	U1	-	numSV	-	Number of satellites used in the position fix	
39	U1	-	reserved2	-	Reserved	

32.7 UBX-LOG-RETRIEVSTRING (0x21 0x0d)

32.7.1 Byte string log entry

Message	LOG-RETRIEVSTRING					
Description	Byte string log entry					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message is used to report a byte string log entry					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x21	0x0d	16 + 1*byteCount	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	entryIndex	-	The index of this log entry	
4	U1	-	version	-	The version of this message. Set to 0	
5	U1	-	reserved1	-	Reserved	
6	U2	-	year	-	Year (1-65635) of UTC time. Will be zero if time not known	
8	U1	-	month	-	Month (1-12) of UTC time	
9	U1	-	day	-	Day (1-31) of UTC time	
10	U1	-	hour	-	Hour (0-23) of UTC time	
11	U1	-	minute	-	Minute (0-59) of UTC time	
12	U1	-	second	-	Second (0-60) of UTC time	
13	U1	-	reserved2	-	Reserved	
14	U2	-	byteCount	-	Size of string in bytes	
Start of repeated block (byteCount times)						
16 + 1*N	U1	-	bytes	-	The bytes of the string	
End of repeated block						

32.8 UBX-LOG-RETRIEVE (0x21 0x09)

32.8.1 Request log data

Message	LOG-RETRIEVE
Description	Request log data
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00
Type	Command
Comment	<p>This message is used to request logged data (log recording must first be disabled, see UBX-CFG-LOGFILTER).</p> <p>Log entries are returned in chronological order, using the messages UBX-LOG-RETRIEVEPOS and UBX-LOG-RETRIEVSTRING. The maximum number of entries that can be returned in response to a single UBX-LOG-RETRIEVE message is 256. If more entries than this are required the message will need to be sent multiple times with different startNumbers. The retrieve will be stopped if any UBX-LOG message is received. The speed of transfer can be maximised by using a high data rate and temporarily stopping the GPS processing (see UBX-CFG-RST)</p>

		Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure		0xB5 0x62	0x21	0x09	12	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	U4	-	startNumber	-	Index of first entry to be transferred		
4	U4	-	entryCount	-	Number of log entries to transfer. The maximum is 256		
8	U1	-	version	-	The version of this message. Set to 0		
9	U1[3]	-	reserved	-	Reserved		

32.9 UBX-LOG-STRING (0x21 0x04)

32.9.1 Store arbitrary string in on-board Flash memory

Message	LOG-STRING						
Description	Store arbitrary string in on-board Flash memory						
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 						
Type	Command						
Comment	This message can be used to store an arbitrary byte string in the on-board flash memory. The maximum length that can be stored is 256 bytes.						
		Header	Class	ID	Length (Bytes)	Payload	Checksum
Message Structure		0xB5 0x62	0x21	0x04	0 + 1*N	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
Start of repeated block (N times)							
N*1	U1	-	bytes	-	The string of bytes to be logged (maximum 256)		
End of repeated block							

33 UBX-MGA (0x13)

Multi-GNSS Assistance: i.e. Assistance data for various GNSS.

33.1 UBX-MGA-ACK (0x13 0x60)

33.1.1 UBX-MGA-ACK-DATA0

Message	UBX-MGA-ACK-DATA0					
Description	Multi-GNSS Acknowledge message					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message is sent by a u-blox receiver to acknowledge the receipt of an assistance message. Acknowledgements are enabled by setting the <code>ackAiding</code> parameter in the UBX-CFG-NAVX5 message. See the description of flow control for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x60	8	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Type, 1 = ACK, 0 = NACK	
1	U1	-	version	-	The version of this message, always set to 0	
2	U1	-	errorCode	-	Indicates the reason why a NACK was returned: 0: No error occurred (only if message type is ACK) 1: The receiver doesn't know the time so can't use the data (To resolve this an UBX-MGA-INI-TIME_UTC message should be supplied first) 2: The message version is not supported by the receiver 3: The message size does not match the message version 4: The message data could not be stored to the database 5: The receiver is not ready to use the message data 6: The message type is unknown 255: Undefined error occurred	
3	U1	-	msgId	-	UBX message ID of the ack'ed message	
4	U1[4]	-	msgPayloadStart	-	The first 4 bytes of the ack'ed message's payload	

33.2 UBX-MGA-ANO (0x13 0x20)

33.2.1 Multi-GNSS AssistNow Offline Assistance

Message	MGA-ANO					
Description	Multi-GNSS AssistNow Offline Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message is created by the AssistNow Offline service to deliver AssistNow Offline assistance to the receiver. See the description of AssistNow Offline for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x20	76	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	message type (always 0x00)	
1	U1	-	version	-	message version (always 0x00)	
2	U1	-	svId	-	Satellite identifier (see Satellite Numbering)	
3	U1	-	gnssId	-	GNSS identifier (see Satellite Numbering)	
4	U1	-	year	-	years since the year 2000	
5	U1	-	month	-	month (1..12)	
6	U1	-	day	-	day (1..31)	
7	U1	-	reserved0	-	reserved	
8	U1[64]	-	data	-	assistance data	
72	U4	-	reserved1	-	reserved	

33.3 UBX-MGA-DBD (0x13 0x80)

33.3.1 Poll the Navigation Database.

Message	MGA-DBD					
Description	Poll the Navigation Database.					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Poll the whole navigation data base. The receiver will send all available data from its internal database. The receiver will indicate the finish of the transmission with a UBX-MGA-ACK . The msgPayloadStart field of the UBX-MGA-ACK message will contain a U4 representing the number of UBX-MGA-DBD-DATA* messages sent.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x80	0	see below	CK_A CK_B
No payload						

33.3.2 Navigation Database Dump Entry

Message	MGA-DBD					
Description	Navigation Database Dump Entry					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input / Output Message					
Comment	<p>UBX-MGA-DBD messages are only intended to be sent back to the same receiver that generated them.</p> <p>Navigation database entry. The data fields are firmware specific. Transmission of this type of message will be acknowledged by MGA-ACK messages, if acknowledgement has been enabled (see the description of flow control for details).</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x80	12 + 1*N	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	reserved0	-	Reserved	
4	U4	-	reserved1	-	Reserved	
8	U4	-	reserved2	-	Reserved	
Start of repeated block (N times)						
12 + 1*N	U1	-	data	-	fw specific data	
End of repeated block						

33.4 UBX-MGA-FLASH (0x13 0x21)

33.4.1 UBX-MGA-FLASH-DATA

Message	UBX-MGA-FLASH-DATA					
Description	Transfer MGA-ANO data block to flash.					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	<p>This message is used to transfer a block of MGA-ANO data from host to the receiver. Upon reception of this message, the receiver will write the payload data to its internal non-volatile memory (flash). Also, on reception of the first MGA-FLASH-DATA message, the receiver will erase the flash allocated to storing any existing MGA-ANO data. The payload can be up to 512 bytes. Payloads larger than this would exceed the receiver's internal buffering capabilities. The receiver will ACK/NACK this message using the message alternatives given below. The host shall wait for an acknowledge message before sending the next data block. See Flash-based AssistNow Offline for details.</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x21	6 + 1*size	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 1 for this message.	
1	U1	-	version	-	FLASH-DATA message version (this is version 0).	

MGA-FLASH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U2	-	sequence	-	Message sequence number, starting at 0 and incrementing by 1 for each MGA-FLASH-DATA message sent.
4	U2	-	size	-	Payload size in bytes.
<i>Start of repeated block (size times)</i>					
6 + 1*N	U1	-	data	-	Payload data.
<i>End of repeated block</i>					

33.4.2 UBX-MGA-FLASH-STOP

Message	UBX-MGA-FLASH-STOP					
Description	Finish flashing MGA-ANO data.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message is used to tell the receiver that there are no more MGA-FLASH type 1 messages coming, and that it can do any final internal operations needed to commit the data to flash as a background activity. A UBX-MGA-ACK message will be sent at the end of this process. Note that there may be a delay of several seconds before the UBX-MGA-ACK for this message is sent because of the time taken for this processing. See Flash-based AssistNow Offline for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x21	2	see below	CK_A CK_B
<i>Payload Contents:</i>						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 2 for this message.	
1	U1	-	version	-	FLASH-STOP message version (this is version 0).	

33.4.3 UBX-MGA-FLASH-ACK

Message	UBX-MGA-FLASH-ACK					
Description	Acknowledge last FLASH-DATA or FLASH-STOP.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message reports an ACK/NACK to the host for the last MGA-FLASH type 1 or type 2 message message received. See Flash-based AssistNow Offline for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x21	6	see below	CK_A CK_B
<i>Payload Contents:</i>						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 3 for this message.	
1	U1	-	version	-	FLASH-ACK message version (this is version 0).	

MGA-FLASH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U1	-	ack	-	Acknowledgement type. 0 - ACK: Message received and written to flash. 1 - NACK: Problem with last message, re-transmission required (this only happens while acknowledging a UBXMGA_FLASH_DATA message). 2 - NACK: problem with last message, give up.
3	U1	-	reserved	-	Reserved.
4	U2	-	sequence	-	If acknowledging a UBXMGA_FLASH_DATA message this is the Message sequence number being ack'ed. If acknowledging a UBXMGA_FLASH_STOP message it will be set to 0xffff.

33.5 UBXMGA-GLO (0x13 0x06)

33.5.1 UBXMGA-GLO-EPH

Message	UBXMGA-GLO-EPH					
Description	GLONASS Ephemeris Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of GLONASS ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x06	48	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 1 for this message (1 = Ephemeris).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U1	-	svId	-	GLONASS Satellite identifier (see Satellite Numbering)	
3	U1	-	reserved2	-	Reserved, set to 0	
4	U1	-	FT	-	User range accuracy	
5	U1	-	B	-	Health flag from string 2	
6	U1	-	M	-	Type of GLONASS satellite (1 indicates GLONASS-M)	
7	I1	-	H	-	Carrier frequency number of navigation RF signal, Range=(-7 .. 6), -128 for unknown	
8	I4	2 ⁻¹¹	x	kilometers	X component of the SV position in PZ-90.02 coordinate System	
12	I4	2 ⁻¹¹	y	kilometers	Y component of the SV position in PZ-90.02 coordinate System	

MGA-GLO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16	I4	2 ⁻¹¹	z	kilometers	Z component of the SV position in PZ-90.02 coordinate System
20	I4	2 ⁻²⁰	dx	kilometers/sec	X component of the SV velocity in PZ-90.02 coordinate System
24	I4	2 ⁻²⁰	dy	kilometers/sec	Y component of the SV velocity in PZ-90.02 coordinate System
28	I4	2 ⁻²⁰	dz	kilometers/sec	Z component of the SV velocity in PZ-90.02 coordinate System
32	I1	2 ⁻³⁰	ddx	kilometers/sec ²	X component of the SV acceleration in PZ-90.02 coordinate System
33	I1	2 ⁻³⁰	ddy	kilometers/sec ²	Y component of the SV acceleration in PZ-90.02 coordinate System
34	I1	2 ⁻³⁰	ddz	kilometers/sec ²	Z component of the SV acceleration in PZ-90.02 coordinate System
35	U1	15	tb	minutes	Index of a time interval within current day according to UTC(SU)
36	I2	2 ⁻⁴⁰	gamma	-	Relative carrier frequency deviation
38	U1	-	E	days	Ephemeris data age indicator
39	I1	2 ⁻³⁰	deltaTau	seconds	Time difference between L2 and L1 band
40	I4	2 ⁻³⁰	tau	seconds	SV clock bias
44	U4	-	reserved3	-	Reserved, set to 0

33.5.2 UBX-MGA-GLO-ALM

Message	UBX-MGA-GLO-ALM					
Description	GLONASS Almanac Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of GLONASS almanac assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x06	36	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 2 for this message (2 = Almanac).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U1	-	svId	-	GLONASS Satellite identifier (see Satellite Numbering)	
3	U1	-	reserved2	-	Reserved, set to 0	

MGA-GLO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	U2	-	N	days	Reference calendar day number of almanac within the four-year period (from string 5)
6	U1	-	M	-	Type of GLONASS satellite (1 indicates GLONASS-M)
7	U1	-	C	-	Unhealthy flag at instant of almanac upload (1 indicates operability of satellite)
8	I2	2 ⁻¹⁸	tau	seconds	Coarse time correction to GLONASS time
10	U2	2 ⁻²⁰	epsilon	-	Eccentricity
12	I4	2 ⁻²⁰	lambda	semi-circles	Longitude of the first (within the N-day) ascending node of satellite orbit in PC-90.02 coordinate system
16	I4	2 ⁻²⁰	deltaI	semi-circles	Correction to the mean value of inclination
20	U4	2 ⁻⁵	tLambda	seconds	Time of the first ascending node passage
24	I4	2 ⁻⁹	deltaT	seconds /orbital-period	Correction to the mean value of Draconian period
28	I1	2 ⁻¹⁴	deltaDT	seconds /orbital-period ²	Rate of change of Draconian period
29	I1	-	H	-	Carrier frequency number of navigation RF signal, Range=(-7 .. 6)
30	I2	-	omega	-	Argument of perigee
32	U4	-	reserved3	-	Reserved, set to 0

33.5.3 UBX-MGA-GLO-TIMEOFFSET

Message	UBX-MGA-GLO-TIMEOFFSET					
Description	GLONASS Auxiliary Time Offset Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of auxiliary GLONASS assistance (including the GLONASS time offsets to other GNSS systems) to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x06	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 3 for this message (3 = time offsets).	
1	U1	-	reserved1	-	Reserved, set to 0	

MGA-GLO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U2	-	N	days	Reference calendar day number within the four-year period of almanac (from string 5)
4	I4	2 ⁻²⁷	tauC	seconds	Time scale correction to UTC(SU) time
8	I4	2 ⁻³¹	tauGps	seconds	Correction to GPS time relative to GLONASS time
12	I2	2 ⁻¹⁰	B1	seconds	Coefficient to determine delta UT1
14	I2	2 ⁻¹⁶	B2	seconds /msd	Rate of change of delta UT1
16	U4	-	reserved2	-	Reserved, set to 0

33.6 UBX-MGA-GPS (0x13 0x00)

33.6.1 UBX-MGA-GPS-EPH

Message	UBX-MGA-GPS-EPH					
Description	GPS Ephemeris Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of GPS ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x00	68	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 1 for this message (1 = Ephemeris).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U1	-	svId	-	GPS Satellite identifier (see Satellite Numbering)	
3	U1	-	reserved2	-	Reserved, set to 0	
4	U1	-	fitInterval	-	Fit interval flag	
5	U1	-	uraIndex	-	URA index	
6	U1	-	svHealth	-	SV health	
7	I1	2 ⁻³¹	tgd	seconds	Group delay differential	
8	U2	-	iodc	-	IODC	
10	U2	2 ⁴	toc	seconds	Clock data reference time	
12	U1	-	reserved3	-	Reserved, set to 0	
13	I1	2 ⁻⁵⁵	af2	sec/sec squared	Time polynomial coefficient 2	
14	I2	2 ⁻⁴³	af1	sec/sec	Time polynomial coefficient 1	
16	I4	2 ⁻³¹	af0	seconds	Time polynomial coefficient 0	
20	I2	2 ⁻⁵	crs	meters	Crs	
22	I2	2 ⁻⁴³	deltaN	semi-circles/sec	Mean motion difference from computed value	

MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
24	I4	2 ⁻³¹	m0	semi-circles	Mean anomaly at reference time
28	I2	2 ⁻²⁹	cuc	radians	Amplitude of cosine harmonic correction term to argument of latitude
30	I2	2 ⁻²⁹	cus	radians	Amplitude of sine harmonic correction term to argument of latitude
32	U4	2 ⁻³³	e	-	Eccentricity
36	U4	2 ⁻¹⁹	sqrtA	sqrt meters	Square root of the semi-major axis
40	U2	2 ⁴	toe	seconds	Reference time of ephemeris
42	I2	2 ⁻²⁹	cic	radians	Amplitude of cos harmonic correction term to angle of inclination
44	I4	2 ⁻³¹	omega0	semi-circles	Longitude of ascending node of orbit plane at weekly epoch
48	I2	2 ⁻²⁹	cis	radians	Amplitude of sine harmonic correction term to angle of inclination
50	I2	2 ⁻⁵	crc	meters	Amplitude of cosine harmonic correction term to orbit radius
52	I4	2 ⁻³¹	i0	semi-circles	Inclination angle at reference time
56	I4	2 ⁻³¹	omega	semi-circles	Argument of perigee
60	I4	2 ⁻⁴³	omegaDot	semi-circles/sec	Rate of right ascension
64	I2	2 ⁻⁴³	idot	semi-circles/sec	Rate of inclination angle
66	U2	-	reserved4	-	Reserved, set to 0

33.6.2 UBX-MGA-GPS-ALM

Message	UBX-MGA-GPS-ALM					
Description	GPS Almanac Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of GPS almanac assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x00	36	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 2 for this message (2 = Almanac).	
1	U1	-	reserved1	-	Reserved, set to 0	

MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U1	-	svId	-	GPS Satellite identifier (see Satellite Numbering)
3	U1	-	svHealth	-	SV health information
4	U2	2 ⁻²¹	e	-	Eccentricity
6	U1	-	almWNa	week	Reference week number of almanac (the 8 bit WNa field)
7	U1	2 ¹²	toa	seconds	Reference time of almanac
8	I2	2 ⁻¹⁹	deltaI	semi-circles	Delta inclination angle at reference time
10	I2	2 ⁻³⁸	omegaDot	semi-circles/sec	Rate of right ascension
12	U4	2 ⁻¹¹	sqrta	sqrt meters	Square root of the semi-major axis
16	I4	2 ⁻²³	omega0	semi-circles	Longitude of ascending node of orbit plane
20	I4	2 ⁻²³	omega	semi-circles	Argument of perigee
24	I4	2 ⁻²³	m0	semi-circles	Mean anomaly at reference time
28	I2	2 ⁻²⁰	af0	seconds	Time polynomial coefficient 0 (8 MSBs)
30	I2	2 ⁻³⁸	af1	sec/sec	Time polynomial coefficient 1
32	U4	-	reserved2	-	Reserved, set to 0

33.6.3 UBX-MGA-GPS-HEALTH

Message	UBX-MGA-GPS-HEALTH					
Description	GPS Health Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of GPS health assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x00	40	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 4 for this message (4 = health flags).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U2	-	reserved2	-	Reserved, set to 0	
4	U1[32]	-	healthCode	-	Each byte represents a GPS SV (1-32). The 6 LSBs of each byte contains the 6 bit health code from subframes 4/5 page 25.	
36	U4	-	reserved3	-	Reserved, set to 0.	

33.6.4 UBX-MGA-GPS-UTC

<i>Message</i>		UBX-MGA-GPS-UTC					
<i>Description</i>		GPS UTC Assistance					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>		Input					
<i>Comment</i>		This message allows the delivery of GPS UTC assistance to a receiver. See the description of AssistNow Online for details.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x13	0x00	20	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		
0	U1	-	type	-	Message type. Set to 5 for this message (5 = Time parameters).		
1	U1	-	reserved1	-	Reserved, set to 0		
2	U2	-	reserved2	-	Reserved, set to 0		
4	I4	2 ⁻³⁰	utcA0	seconds	First parameter of UTC polynomial		
8	I4	2 ⁻⁵⁰	utcA1	sec/sec	Second parameter of UTC polynomial		
12	I1	-	utcDtLS	seconds	Delta time due to current leap seconds		
13	U1	2 ¹²	utcTot	seconds	UTC parameters reference time of week (GPS time)		
14	U1	-	utcWNt	weeks	UTC parameters reference week number (the 8 bit WNt field)		
15	U1	-	utcWNlSF	weeks	Week number at the end of which the future leap second becomes effective (the 8 bit WNLSF field)		
16	U1	-	utcDn	days	Day number at the end of which the future leap second becomes effective		
17	I1	-	utcDtLSF	seconds	Delta time due to future leap seconds		
18	U2	-	reserved3	-	Reserved, set to 0		

33.6.5 UBX-MGA-GPS-IONO

<i>Message</i>		UBX-MGA-GPS-IONO					
<i>Description</i>		GPS Ionosphere Assistance					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>		Input					
<i>Comment</i>		This message allows the delivery of GPS ionospheric assistance to a receiver. See the description of AssistNow Online for details.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x13	0x00	16	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		

MGA-GPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type. Set to 6 for this message (6 = ionosphere parameters).
1	U1	-	reserved1	-	Reserved, set to 0
2	U2	-	reserved2	-	Reserved, set to 0
4	I1	2 ⁻³⁰	ionoAlpha0	seconds	Ionospheric parameter alpha0 [s]
5	I1	2 ⁻²⁷	ionoAlpha1	sec/semi-circle	Ionospheric parameter alpha1 [s/semi-circle]
6	I1	2 ⁻²⁴	ionoAlpha2	sec/(semi-circle ²)	Ionospheric parameter alpha2 [s/semi-circle ²]
7	I1	2 ⁻²⁴	ionoAlpha3	sec/(semi-circle ³)	Ionospheric parameter alpha3 [s/semi-circle ³]
8	I1	2 ¹¹	ionoBeta0	seconds	Ionospheric parameter beta0 [s]
9	I1	2 ¹⁴	ionoBeta1	sec/semi-circle	Ionospheric parameter beta1 [s/semi-circle]
10	I1	2 ¹⁶	ionoBeta2	sec/(semi-circle ²)	Ionospheric parameter beta2 [s/semi-circle ²]
11	I1	2 ¹⁶	ionoBeta3	sec/(semi-circle ³)	Ionospheric parameter beta3 [s/semi-circle ³]
12	U4	-	reserved3	-	Reserved, set to 0

33.7 UBX-MGA-INI (0x13 0x40)

33.7.1 UBX-MGA-INI-POS_XYZ

Message	UBX-MGA-INI-POS_XYZ					
Description	Initial Position Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	<p>Supplying position assistance that is inaccurate by more than the specified position accuracy, may lead to substantially degraded receiver performance.</p> <p>This message allows the delivery of initial position assistance to a receiver in cartesian ECEF coordinates. This message is equivalent to the UBX-MGA-INI-POS_LLH message, except for the coordinate system. See the description of AssistNow Online for details.</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 0x00 for this message (0x00 = Position - ECEF - XYZ).	
1	U1	-	reserved1	-	Reserved, set to 0	

MGA-INI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
2	U2	-	reserved2	-	Reserved, set to 0
4	I4	-	ecefX	cm	WGS84 ECEF X coordinate
8	I4	-	ecefY	cm	WGS84 ECEF Y coordinate
12	I4	-	ecefZ	cm	WGS84 ECEF Z coordinate
16	U4	-	posAcc	cm	Position accuracy (stddev)

33.7.2 UBX-MGA-INI-POS_LLH

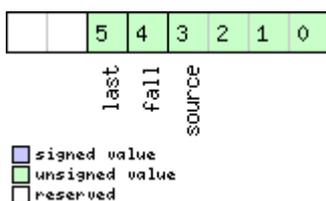
Message	UBX-MGA-INI-POS_LLH					
Description	Initial Position Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	<p>Supplying position assistance that is inaccurate by more than the specified position accuracy, may lead to substantially degraded receiver performance.</p> <p>This message allows the delivery of initial position assistance to a receiver in WGS84 lat/long/alt coordinates. This message is equivalent to the UBX-MGA-INI-POS_XYZ message, except for the coordinate system. See the description of AssistNow Online for details.</p>					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 0x01 for this message (0x01 = Position - ECEF - LLA).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U2	-	reserved2	-	Reserved, set to 0	
4	I4	1e-7	lat	deg	WGS84 Latitude	
8	I4	1e-7	lon	deg	WGS84 Longitude	
12	I4	-	alt	cm	WGS84 Altitude	
16	U4	-	posAcc	cm	Position accuracy (stddev)	

33.7.3 UBX-MGA-INI-TIME.UTC

Message		UBX-MGA-INI-TIME.UTC					
Description		Initial Time Assistance					
Firmware		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type		Input					
Comment		Supplying time assistance that is inaccurate by more than the specified time accuracy, may lead to substantially degraded receiver performance. This message allows the delivery of UTC time assistance to a receiver. This message is equivalent to the UBX-MGA-INI-TIME_GNSS message, except for the time base. See the description of AssistNow Online for details.					
Message Structure		Header	Class	ID	Length (Bytes)	Payload	Checksum
		0xB5 0x62	0x13	0x40	24	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	U1	-	type	-	Message type. Set to 0x10 for this message (0x10 = Time).		
1	U1	-	reserved1	-	Reserved, set to 0		
2	X1	-	ref	-	Reference to be used to set time (see graphic below)		
3	I1	-	leapSecs	s	Number of leap seconds since 1980 (or 0x80 = -128 if unknown)		
4	U2	-	year	-	Year		
6	U1	-	month	-	Month, starting at 1		
7	U1	-	day	-	Day, starting at 1		
8	U1	-	hour	-	Hour, from 0 to 23		
9	U1	-	minute	-	Minute, from 0 to 59		
10	U1	-	second	s	Seconds, from 0 to 59		
11	U1	-	reserved2	-	Reserved, set to 0		
12	U4	-	ns	ns	Nanoseconds, from 0 to 999,999,999		
16	U2	-	tAccS	s	Seconds part of time accuracy		
18	U2	-	reserved3	-	Reserved, set to 0		
20	U4	-	tAccNs	ns	Nanoseconds part of time accuracy, from 0 to 999,999,999		

Bitfield ref

This Graphic explains the bits of ref



Name	Description
------	-------------

Bitfield ref Description continued

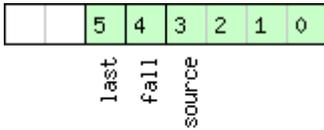
Name	Description
source	0: none, i.e. on receipt of message (will be inaccurate!) 1: relative to pulse sent to EXTINT0 2: relative to pulse sent to EXTINT1 3-15: reserved
fall	use falling edge of EXTINT pulse (default rising) - only if source is EXTINT
last	use last EXTINT pulse (default next pulse) - only if source is EXTINT

33.7.4 UBX-MGA-INI-TIME_GNSS

Message	UBX-MGA-INI-TIME_GNSS					
Description	Initial Time Assistance					
Firmware	Supported on: • u-blox M8 firmware version 2.00					
Type	Input					
Comment	Supplying time assistance that is inaccurate by more than the specified time accuracy, may lead to substantially degraded receiver performance. This message allows the delivery of time assistance to a receiver in a chosen GNSS timebase. This message is equivalent to the UBX-MGA-INI-TIME_UTC message, except for the time base. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	24	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 0x11 for this message (0x11 = Time GNSS).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	X1	-	ref	-	Reference to be used to set time (see graphic below)	
3	U1	-	gnssId	-	Source of time information. Currently supported: 0: GPS time 2: Galileo time 3: BeiDou time 6: GLONASS time: week = 834 + ((N4-1)*1461 + Nt)/7, tow = (((N4-1)*1461 + Nt) % 7) * 86400 + tod	
4	U2	-	reserved2	-	Reserved, set to 0	
6	U2	-	week	-	GNSS week number	
8	U4	-	tow	-	GNSS time of week	
12	U4	-	ns	-	GNSS time of week, nanosecond part from 0 to 999,999,999	
16	U2	-	tAccS	s	Seconds part of time accuracy	
18	U2	-	reserved3	-	Reserved, set to 0	
20	U4	-	tAccNs	ns	Nanoseconds part of time accuracy, from 0 to 999,999,999	

Bitfield ref

This Graphic explains the bits of `ref`



■ signed value
■ unsigned value
■ reserved

Name	Description
source	0: none, i.e. on receipt of message (will be inaccurate!) 1: relative to pulse sent to EXTINT0 2: relative to pulse sent to EXTINT1 3-15: reserved
fall	use falling edge of EXTINT pulse (default rising) - only if source is EXTINT
last	use last EXTINT pulse (default next pulse) - only if source is EXTINT

33.7.5 UBX-MGA-INI-CLKD

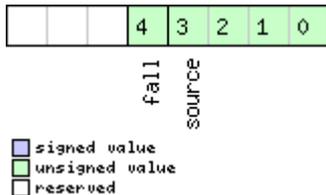
Message	UBX-MGA-INI-CLKD					
Description	Initial Clock Drift Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	Supplying clock drift assistance that is inaccurate by more than the specified accuracy, may lead to substantially degraded receiver performance. This message allows the delivery of clock drift assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	12	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 0x20 for this message (0x20 = Clock Drift).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U2	-	reserved2	-	Reserved, set to 0	
4	I4	-	clkD	ns/s	Clock drift	
8	U4	-	clkDAcc	ns/s	Clock drift accuracy	

33.7.6 UBX-MGA-INI-FREQ

Message	UBX-MGA-INI-FREQ					
Description	Initial Frequency Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	Supplying external frequency assistance that is inaccurate by more than the specified accuracy, may lead to substantially degraded receiver performance. This message allows the delivery of external frequency assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x40	12	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 0x21 for this message (0x21 = Frequency).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U1	-	reserved2	-	Reserved, set to 0	
3	X1	-	flags	-	Frequency reference (see graphic below)	
4	I4	1e-2	freq	Hz	Frequency	
8	U4	-	freqAcc	ppb	Frequency accuracy	

Bitfield flags

This Graphic explains the bits of flags



Name	Description
source	0: frequency available on EXTINT0 1: frequency available on EXTINT1 2-15: reserved
fall	use falling edge of EXTINT pulse (default rising)

33.7.7 UBX-MGA-INI-EOP

<i>Message</i>		UBX-MGA-INI-EOP					
<i>Description</i>		Earth Orientation Parameters Assistance					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>		Input					
<i>Comment</i>		This message allows the delivery of new Earth Orientation Parameters (EOP) to a receiver to improve AssistNow Autonomous operation.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x13	0x40	72	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		
0	U1	-	type	-	Message type. Set to 0x30 for this message (0x30 = EOP).		
1	U1	-	reserved1	-	Reserved, set to 0		
2	U2	-	reserved2	-	Reserved, set to 0		
4	U2	-	d2kRef	d	reference time (days since 1.1.2000 12.00h UTC)		
6	U2	-	d2kMax	d	expiration time (days since 1.1.2000 12.00h UTC)		
8	I4	2 ⁻³⁰	xpP0	arcsec	x _p t ⁰ polynomial term (offset)		
12	I4	2 ⁻³⁰	xpP1	arcsec/d	x _p t ¹ polynomial term (drift)		
16	I4	2 ⁻³⁰	ypP0	arcsec	y _p t ⁰ polynomial term (offset)		
20	I4	2 ⁻³⁰	ypP1	arcsec/d	y _p t ¹ polynomial term (drift)		
24	I4	2 ⁻²⁵	dUT1	s	dUT1 t ⁰ polynomial term (offset)		
28	I4	2 ⁻³⁰	ddUT1	s/d	dUT1 t ¹ polynomial term (drift)		
32	U4[10]	-	reserved3	-	Reserved, set all to 0		

33.8 UBX-MGA-QZSS (0x13 0x05)

33.8.1 UBX-MGA-QZSS-EPH

<i>Message</i>		UBX-MGA-QZSS-EPH					
<i>Description</i>		QZSS Ephemeris Assistance					
<i>Firmware</i>		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>		Input					
<i>Comment</i>		This message allows the delivery of QZSS ephemeris assistance to a receiver. See the description of AssistNow Online for details.					
<i>Message Structure</i>		<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
		0xB5 0x62	0x13	0x05	68	see below	CK_A CK_B
<i>Payload Contents:</i>							
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>		

MGA-QZSS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U1	-	type	-	Message type. Set to 1 for this message (1 = Ephemeris).
1	U1	-	reserved1	-	Reserved, set to 0
2	U1	-	svId	-	QZSS Satellite identifier (see Satellite Numbering), Range 1-5
3	U1	-	reserved2	-	Reserved, set to 0
4	U1	-	fitInterval	-	Fit interval flag
5	U1	-	uraIndex	-	URA index
6	U1	-	svHealth	-	SV health
7	I1	2 ⁻³¹	tgd	seconds	Group delay differential
8	U2	-	iodc	-	IODC
10	U2	2 ⁴	toc	seconds	Clock data reference time
12	U1	-	reserved3	-	Reserved, set to 0
13	I1	2 ⁻⁵⁵	af2	sec/sec squared	Time polynomial coefficient 2
14	I2	2 ⁻⁴³	af1	sec/sec	Time polynomial coefficient 1
16	I4	2 ⁻³¹	af0	seconds	Time polynomial coefficient 0
20	I2	2 ⁻⁵	crs	meters	Crs
22	I2	2 ⁻⁴³	deltaN	semi-circles/sec	Mean motion difference from computed value
24	I4	2 ⁻³¹	m0	semi-circles	Mean anomaly at reference time
28	I2	2 ⁻²⁹	cuc	radians	Amp of cosine harmonic corr term to arg of lat
30	I2	2 ⁻²⁹	cus	radians	Amp of sine harmonic corr term to arg of lat
32	U4	2 ⁻³³	e	-	eccentricity
36	U4	2 ⁻¹⁹	sqrtA	sqrt meters	Square root of the semi-major axis A
40	U2	2 ⁴	toe	seconds	Reference time of ephemeris
42	I2	2 ⁻²⁹	cic	radians	Amp of cos harmonic corr term to angle of inclination
44	I4	2 ⁻³¹	omega0	semi-circles	Long of asc node of orbit plane at weekly epoch
48	I2	2 ⁻²⁹	cis	radians	Amp of sine harmonic corr term to angle of inclination
50	I2	2 ⁻⁵	crc	meters	Amp of cosine harmonic corr term to orbit radius
52	I4	2 ⁻³¹	i0	semi-circles	Inclination angle at reference time
56	I4	2 ⁻³¹	omega	semi-circles	Argument of perigee
60	I4	2 ⁻⁴³	omegaDot	semi-circles/sec	Rate of right ascension
64	I2	2 ⁻⁴³	idot	semi-circles/sec	Rate of inclination angle
66	U2	-	reserved4	-	Reserved, set to 0

33.8.2 UBX-MGA-QZSS-ALM

Message	UBX-MGA-QZSS-ALM					
Description	QZSS Almanac Assistance					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Input					
Comment	This message allows the delivery of QZSS almanac assistance to a receiver. See the description of AssistNow Online for details.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x13	0x05	36	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	type	-	Message type. Set to 2 for this message (2 = Almanac).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U1	-	svId	-	QZSS Satellite identifier (see Satellite Numbering), Range 1-5	
3	U1	-	svHealth	-	Almanac SV health information	
4	U2	2 ⁻²¹	e	-	Almanac eccentricity	
6	U1	-	almWNa	week	Reference week number of almanac (the 8 bit WNa field)	
7	U1	2 ¹²	toa	seconds	Reference time of almanac	
8	I2	2 ⁻¹⁹	deltaI	semi-circles	Delta inclination angle at reference time	
10	I2	2 ⁻³⁸	omegaDot	semi-circles/sec	Almanac rate of right ascension	
12	U4	2 ⁻¹¹	sqrTA	sqrt meters	Almanac square root of the semi-major axis A	
16	I4	2 ⁻²³	omega0	semi-circles	Almanac long of asc node of orbit plane at weekly	
20	I4	2 ⁻²³	omega	semi-circles	Almanac argument of perigee	
24	I4	2 ⁻²³	m0	semi-circles	Almanac mean anomaly at reference time	
28	I2	2 ⁻²⁰	af0	seconds	Almanac time polynomial coefficient 0 (8 MSBs)	
30	I2	2 ⁻³⁸	af1	sec/sec	Almanac time polynomial coefficient 1	
32	U4	-	reserved2	-	Reserved, set to 0	

33.8.3 UBX-MGA-QZSS-HEALTH

<i>Message</i>	UBX-MGA-QZSS-HEALTH					
<i>Description</i>	QZSS Health Assistance					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Input					
<i>Comment</i>	This message allows the delivery of QZSS health assistance to a receiver. See the description of AssistNow Online for details.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x13	0x05	12	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	type	-	Message type. Set to 4 for this message (4 = health flags).	
1	U1	-	reserved1	-	Reserved, set to 0	
2	U2	-	reserved2	-	Reserved, set to 0	
4	U1[5]	-	healthCode	-	Each byte represents a QZSS SV (1-5). The 6 LSBs of each byte contains the 6 bit health code from subframes 4/5, data ID = 3, SV ID = 51	
9	U1[3]	-	reserved3	-	Reserved, set to 0	

34 UBX-MON (0x0A)

Monitoring Messages: i.e. Communication 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.

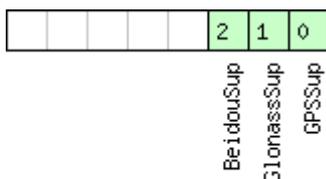
34.1 UBX-MON-GNSS (0x0A 0x28)

34.1.1 Information message GNSS selection

Message	MON-GNSS					
Description	Information message GNSS selection					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Output					
Comment	This message reports GNSS system selection. It does this by means of bit masks in U1 fields. Each bit in a bit mask corresponds to one GNSS system. Systems such as SBAS and QZSS are not reported. If systems such as SBAS/QZSS are related to one GNSS system (GPS is these cases), then they will be disabled when the related system is disabled.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x28	8	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	Type of the message, 1 for this type	
1	X1	-	Supported	-	A bit mask, saying which GNSS systems can be supported by this receiver (see graphic below)	
2	X1	-	Default	-	A bit mask, saying which GNSS systems are enabled in the current efuse default configuration for this receiver (see graphic below)	
3	X1	-	Enabled	-	A bit mask, saying which GNSS systems are currently enabled for this receiver (see graphic below)	
4	U1	-	Simultaneous	-	Maximum number of concurrent GNSS systems which can be supported by this receiver	
5	U1[3]	-	Reserved4	-	reserved	

Bitfield Supported

This Graphic explains the bits of Supported



signed value
 unsigned value
 reserved

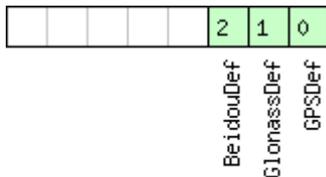
Name	Description
GPSSup	GPS is supported
GlonassSup	GLONASS is supported

Bitfield Supported Description continued

Name	Description
BeidouSup	BeiDou is supported

Bitfield Default

This Graphic explains the bits of Default

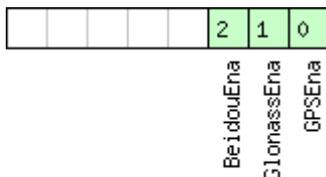


■ signed value
■ unsigned value
■ reserved

Name	Description
GPSDef	GPS is default-enabled
GlonassDef	GLONASS is default-enabled
BeidouDef	BeiDou is default-enabled

Bitfield Enabled

This Graphic explains the bits of Enabled



■ signed value
■ unsigned value
■ reserved

Name	Description
GPSEna	GPS is enabled
GlonassEna	GLONASS is enabled
BeidouEna	BeiDou is enabled

34.2 UBX-MON-HW2 (0x0A 0x0B)

34.2.1 Extended Hardware Status

Message	MON-HW2					
Description	Extended Hardware Status					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
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: <ul style="list-style-type: none"> The smaller the absolute value of the variable $ofsI$ and $ofsQ$, the better. Ideally, the magnitude of the I-part ($magI$) and the Q-part ($magQ$) of the complex signal should be the same. 					
	Header	Class	ID	Length (Bytes)	Payload	Checksum

<i>Message Structure</i>	0xB5 0x62	0x0A	0x0B	28	<i>see below</i>	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
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	U4	-	lowLevCfg	-	Low-level configuration	
12	U4[2]	-	reserved1	-	Reserved	
20	U4	-	postStatus	-	POST status word	
24	U4	-	reserved2	-	Reserved	

34.3 UBX-MON-HW (0x0A 0x09)

34.3.1 Hardware Status

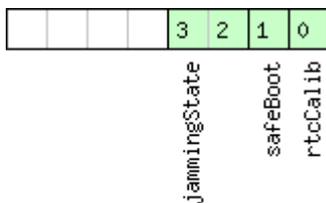
<i>Message</i>	MON-HW					
<i>Description</i>	Hardware Status					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Periodic/Polled					
<i>Comment</i>	Status of different aspect of the hardware, such as Antenna, PIO/Peripheral Pins, Noise Level, Automatic Gain Control (AGC)					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x0A	0x09	60	<i>see below</i>	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
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)	

MON-HW continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
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[17]	-	VP	-	Array of Pin Mappings for each of the 17 Physical Pins
45	U1	-	jamInd	-	CW Jamming indicator, scaled (0 = no CW jamming, 255 = strong CW jamming)
46	U2	-	reserved3	-	Reserved
48	X4	-	pinIrq	-	Mask of Pins Value using the PIO Irq
52	X4	-	pullH	-	Mask of Pins Value using the PIO Pull High Resistor
56	X4	-	pullL	-	Mask of Pins Value using the PIO Pull Low Resistor

Bitfield flags

This Graphic explains the bits of flags



- signed value
- unsigned value
- reserved

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)

34.4 UBX-MON-IO (0x0A 0x02)

34.4.1 I/O Subsystem Status

Message		MON-IO					
Description		I/O Subsystem Status					
Firmware		Supported on: • u-blox M8 firmware version 2.00					
Type		Periodic/Polled					
Comment		The size of the message is determined by the number of ports 'N' the receiver supports, i.e. on u-blox 5 the number of ports is 6.					
Message Structure		Header	Class	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							

34.5 UBX-MON-MSGPP (0x0A 0x06)

34.5.1 Message Parse and Process Status

Message		MON-MSGPP					
Description		Message Parse and Process Status					
Firmware		Supported on: • u-blox M8 firmware version 2.00					
Type		Periodic/Polled					
Comment		-					
Message Structure		Header	Class	ID	Length (Bytes)	Payload	Checksum
		0xB5 0x62	0x0A	0x06	120	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	U2[8]	-	msg1	msgs	Number of successfully parsed messages for each protocol on port0		
16	U2[8]	-	msg2	msgs	Number of successfully parsed messages for each protocol on port1		

MON-MSGPP continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
32	U2[8]	-	msg3	msgs	Number of successfully parsed messages for each protocol on port2
48	U2[8]	-	msg4	msgs	Number of successfully parsed messages for each protocol on port3
64	U2[8]	-	msg5	msgs	Number of successfully parsed messages for each protocol on port4
80	U2[8]	-	msg6	msgs	Number of successfully parsed messages for each protocol on port5
96	U4[6]	-	skipped	bytes	Number skipped bytes for each port

34.6 UBX-MON-PATCH (0x0A 0x27)

34.6.1 Poll Request for installed patches

Message	MON-PATCH					
Description	Poll Request for installed patches					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x27	0	see below	CK_A CK_B
No payload						

34.6.2 Output information about installed patches.

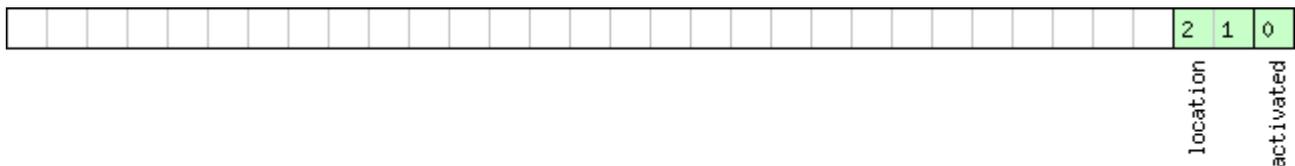
Message	MON-PATCH					
Description	Output information about installed patches.					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output Message					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x27	4 + 16*nEntries	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U2	-	version	-	Type of the message. 0x1 for this one.	
2	U2	-	nEntries	-	The number of patches that is output.	
Start of repeated block (nEntries times)						
4 + 16*N	X4	-	patchInfo	-	Additional information about the patch not stated in the patch header. (see graphic below)	
8 + 16*N	U4	-	comparatorNumber	-	The number of the comparator.	
12 + 16*N	U4	-	patchAddress	-	The address that the targeted by the patch.	

MON-PATCH continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16 + 16*N	U4	-	patchData	-	The data that will be inserted at the patchAddress.
End of repeated block					

Bitfield patchInfo

This Graphic explains the bits of patchInfo



- signed value
- unsigned value
- reserved

Name	Description
activated	1: the patch is active. 0: otherwise.
location	Indicates where the patch is stored. 0: eFuse, 1: ROM, 2: BBR, 3: file system.

34.7 UBX-MON-RXBUF (0x0A 0x07)

34.7.1 Receiver Buffer Status

Message	MON-RXBUF					
Description	Receiver Buffer Status					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	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	

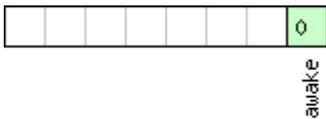
34.8 UBX-MON-RXR (0x0A 0x21)

34.8.1 Receiver Status Information

Message	MON-RXR					
Description	Receiver Status Information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Output					
Comment	The receiver ready message is sent when the receiver changes from or to backup mode.					
Message Structure	Header	Class	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	X1	-	flags	-	Receiver status flags (see graphic below)	

Bitfield flags

This Graphic explains the bits of flags



■ signed value
■ unsigned value
 reserved

Name	Description
awake	not in Backup mode

34.9 UBX-MON-TXBUF (0x0A 0x08)

34.9.1 Transmitter Buffer Status

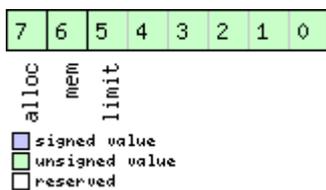
Message	MON-TXBUF					
Description	Transmitter Buffer Status					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x08	28	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U2[6]	-	pending	bytes	Number of bytes pending in transmitter buffer for each target	
12	U1[6]	-	usage	%	Maximum usage transmitter buffer during the last sysmon period for each target	
18	U1[6]	-	peakUsage	%	Maximum usage transmitter buffer for each target	

MON-TXBUF continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
24	U1	-	tUsage	%	Maximum usage of transmitter buffer during the last sysmon period for all targets
25	U1	-	tPeakusage	%	Maximum usage of transmitter buffer for all targets
26	X1	-	errors	-	Error bitmask (see graphic below)
27	U1	-	reserved1	-	Reserved

Bitfield errors

This Graphic explains the bits of errors



Name	Description
limit	Buffer limit of corresponding target reached
mem	Memory Allocation error
alloc	Allocation error (TX buffer full)

34.10 UBX-MON-VER (0x0A 0x04)

34.10.1 Poll Receiver/Software Version

Message	MON-VER					
Description	Poll Receiver/Software Version					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0A	0x04	0	see below	CK_A CK_B
No payload						

34.10.2 Receiver/Software Version

Message		MON-VER					
Description		Receiver/Software Version					
Firmware		Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type		Answer to Poll					
Comment		-					
Message Structure		Header	Class	ID	Length (Bytes)	Payload	Checksum
		0xB5 0x62	0x0A	0x04	40 + 30*N	see below	CK_A CK_B
Payload Contents:							
Byte Offset	Number Format	Scaling	Name	Unit	Description		
0	CH[30]	-	swVersion	-	Zero-terminated Software Version String.		
30	CH[10]	-	hwVersion	-	Zero-terminated Hardware Version String		
Start of repeated block (N times)							
40 + 30*N	CH[30]	-	extension	-	Extended receiver/software information. If the receiver's firmware is running from flash, the first extension field will contain the Software Version String of the underlying ROM. Additional fields may also indicate the supported protocol version and any product variants, capabilities or extensions.		
End of repeated block							

35 UBX-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.

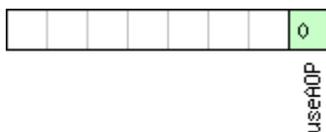
35.1 UBX-NAV-AOPSTATUS (0x01 0x60)

35.1.1 AssistNow Autonomous Status

Message	NAV-AOPSTATUS					
Description	AssistNow Autonomous Status					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message provides information on the status of the <i>AssistNow Autonomous</i> subsystem on the receiver. For example, a host application can determine the optimal time to shut down the receiver by monitoring the <code>status</code> field for a steady 0. See the chapter AssistNow Autonomous in the receiver description for details on this feature.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x60	16	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U1	-	aopCfg	-	<i>AssistNow Autonomous</i> configuration (see graphic below)	
5	U1	-	status	-	<i>AssistNow Autonomous</i> subsystem is idle (0) or running (not 0)	
6	U1	-	reserved0	-	reserved	
7	U1	-	reserved1	-	reserved	
8	U4	-	reserved2	-	reserved	
12	U4	-	reserved3	-	reserved	

Bitfield aopCfg

This Graphic explains the bits of aopCfg



■ signed value
■ unsigned value
■ reserved

Name	Description
useAOP	AOP enabled flag

35.2 UBX-NAV-CLOCK (0x01 0x22)

35.2.1 Clock Solution

Message	NAV-CLOCK					
Description	Clock Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	-					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x22	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	clkB	ns	Clock bias	
8	I4	-	clkD	ns/s	Clock drift	
12	U4	-	tAcc	ns	Time accuracy estimate	
16	U4	-	fAcc	ps/s	Frequency accuracy estimate	

35.3 UBX-NAV-DGPS (0x01 0x31)

35.3.1 DGPS Data Used for NAV

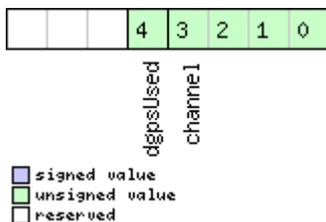
Message	NAV-DGPS					
Description	DGPS Data Used for NAV					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message outputs the DGPS correction data that has been applied to the current NAV Solution. See also the notes on the RTCM protocol .					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x31	16 + 12*numCh	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	age	ms	Age of newest correction data	
8	I2	-	baseId	-	DGPS base station identifier	
10	I2	-	baseHealth	-	DGPS base station health status	
12	U1	-	numCh	-	Number of channels for which correction data is following	
13	U1	-	status	-	DGPS correction type status: 0x00: none 0x01: PR+PRR correction	
14	U2	-	reserved1	-	Reserved	

NAV-DGPS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
<i>Start of repeated block (numCh times)</i>					
16 + 12*N	U1	-	svid	-	Satellite ID
17 + 12*N	X1	-	flags	-	Channel number and usage (see graphic below)
18 + 12*N	U2	-	ageC	ms	Age of latest correction data
20 + 12*N	R4	-	prc	m	Pseudorange correction
24 + 12*N	R4	-	prrc	m/s	Pseudorange rate correction
<i>End of repeated block</i>					

Bitfield flags

This Graphic explains the bits of flags



Name	Description
channel1	GPS channel number this SV is on
dgpsUsed	1 = DGPS used for this SV

35.4 UBX-NAV-DOP (0x01 0x04)

35.4.1 Dilution of precision

Message	NAV-DOP					
Description	Dilution of precision					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	<ul style="list-style-type: none"> 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	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x04	18	see below	CK_A CK_B
<i>Payload Contents:</i>						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U2	0.01	gDOP	-	Geometric DOP	
6	U2	0.01	pDOP	-	Position DOP	
8	U2	0.01	tDOP	-	Time DOP	
10	U2	0.01	vDOP	-	Vertical DOP	
12	U2	0.01	hDOP	-	Horizontal DOP	
14	U2	0.01	nDOP	-	Northing DOP	

NAV-DOP continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
16	U2	0.01	eDOP	-	Easting DOP

35.5 UBX-NAV-ODO (0x01 0x09)

35.5.1 Odometer Solution

Message	NAV-ODO					
Description	Odometer Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message outputs the traveled distance since last reset (see NAV-RESETODO) together with an associated estimated accuracy and the total cumulated ground distance (can only be reset by a cold start of the receiver).					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x09	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	version	-	Message version (0 for this version)	
1	U1	-	reserved1	-	Always set to zero	
2	U2	-	reserved2	-	Always set to zero	
4	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
8	U4	-	distance	m	Ground distance since last reset	
12	U4	-	totalDistance	m	Total cumulative ground distance	
16	U4	-	distanceStd	m	Ground distance accuracy (1-sigma)	

35.6 UBX-NAV-ORB (0x01 0x34)

35.6.1 GNSS Orbit Database Info

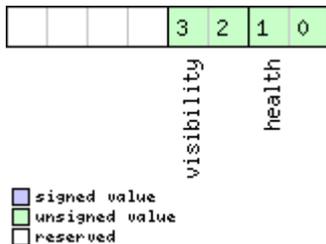
Message	NAV-ORB					
Description	GNSS Orbit Database Info					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	Status of the GNSS orbit database knowledge.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x34	8 + 6*numSv	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U1	-	version	-	Message version (0, for this version)	
5	U1	-	numSv	-	Number of SVs in the database	

NAV-ORB continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
6	U2	-	reserved	-	Reserved
<i>Start of repeated block (numSv times)</i>					
8 + 6*N	U1	-	gnssId	-	GNSS ID
9 + 6*N	U1	-	svId	-	Satellite ID
10 + 6*N	X1	-	svFlag	-	Information Flags (see graphic below)
11 + 6*N	X1	-	eph	-	Ephemeris data (see graphic below)
12 + 6*N	X1	-	alm	-	Almanac data (see graphic below)
13 + 6*N	X1	-	otherOrb	-	Other orbit data available (see graphic below)
<i>End of repeated block</i>					

Bitfield svFlag

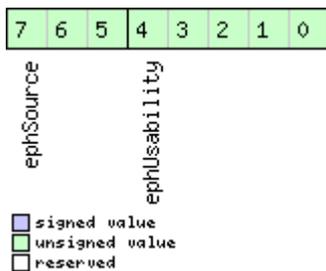
This Graphic explains the bits of svFlag



Name	Description
health	SV health: 0: unknown 1: healthy 2: not healthy
visibility	SV health: 0: unknown 1: below horizon 2: above horizon 3: above elevation mask

Bitfield eph

This Graphic explains the bits of eph



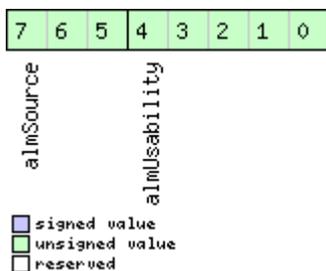
Name	Description
------	-------------

Bitfield eph Description continued

Name	Description
ephUsability	How long the receiver will be able to use the stored ephemeris data from now on: 31: The usability period is unknown 30: The usability period is more than 450 minutes 30 > n > 0: The usability period is between (n-1)*15 and n*15 minutes 0: Ephemeris can no longer be used
ephSource	0: not available 1: GNSS transmission 2: external aiding 3-7: other

Bitfield alm

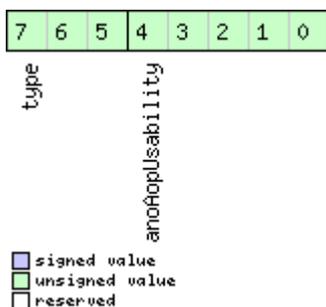
This Graphic explains the bits of a1m



Name	Description
almUsability	How long the receiver will be able to use the stored almanac data from now on: 31: The usability period is unknown 30: The usability period is more than 30 days 30 > n > 0: The usability period is between n-1 and n days 0: Almanac can no longer be used
almSource	0: not available 1: GNSS transmission 2: external aiding 3-7: other

Bitfield otherOrb

This Graphic explains the bits of otherOrb



Name	Description
------	-------------

Bitfield otherOrb Description continued

Name	Description
anoAopUsability	How long the receiver will be able to use the orbit data from now on: 31: The usability period is unknown 30: The usability period is more than 30 days 30 > n > 0: The usability period is between n-1 and n days 0: Data can no longer be used
type	Type of orbit data: 0: No orbit data available 1: Assist now offline data 2: Assist now autonomous data 3-7: Other orbit data

35.7 UBX-NAV-POSECEF (0x01 0x01)

35.7.1 Position Solution in ECEF

Message	NAV-POSECEF					
Description	Position Solution in ECEF					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	See important comments concerning validity of position given in section Navigation Output Filters. -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x01	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	ecefX	cm	ECEF X coordinate	
8	I4	-	ecefY	cm	ECEF Y coordinate	
12	I4	-	ecefZ	cm	ECEF Z coordinate	
16	U4	-	pAcc	cm	Position Accuracy Estimate	

35.8 UBX-NAV-POSLLH (0x01 0x02)

35.8.1 Geodetic Position Solution

Message	NAV-POSLLH					
Description	Geodetic Position Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
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	Class	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 time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	1e-7	lon	deg	Longitude	
8	I4	1e-7	lat	deg	Latitude	
12	I4	-	height	mm	Height above ellipsoid	
16	I4	-	hMSL	mm	Height above mean sea level	
20	U4	-	hAcc	mm	Horizontal accuracy estimate	
24	U4	-	vAcc	mm	Vertical accuracy estimate	

35.9 UBX-NAV-PVT (0x01 0x07)

35.9.1 Navigation Position Velocity Time Solution

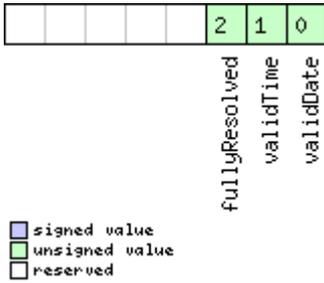
Message	NAV-PVT					
Description	Navigation Position Velocity Time Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	Note that during a leap second there may be more (or less) than 60 seconds in a minute; see the description of leap seconds for details. This message combines position, velocity and time solution, including accuracy figures					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x07	92	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U2	-	year	y	Year (UTC)	
6	U1	-	month	month	Month, range 1..12 (UTC)	
7	U1	-	day	d	Day of month, range 1..31 (UTC)	

NAV-PVT continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
8	U1	-	hour	h	Hour of day, range 0..23 (UTC)
9	U1	-	min	min	Minute of hour, range 0..59 (UTC)
10	U1	-	sec	s	Seconds of minute, range 0..60 (UTC)
11	X1	-	valid	-	Validity Flags (see graphic below)
12	U4	-	tAcc	ns	Time accuracy estimate (UTC)
16	I4	-	nano	ns	Fraction of second, range -1e9 .. 1e9 (UTC)
20	U1	-	fixType	-	GNSSfix Type, range 0..5 0x00 = No Fix 0x01 = Dead Reckoning only 0x02 = 2D-Fix 0x03 = 3D-Fix 0x04 = GNSS + dead reckoning combined 0x05 = Time only fix 0x06..0xff: reserved
21	X1	-	flags	-	Fix Status Flags (see graphic below)
22	U1	-	reserved1	-	Reserved
23	U1	-	numSV	-	Number of satellites used in Nav Solution
24	I4	1e-7	lon	deg	Longitude
28	I4	1e-7	lat	deg	Latitude
32	I4	-	height	mm	Height above ellipsoid
36	I4	-	hMSL	mm	Height above mean sea level
40	U4	-	hAcc	mm	Horizontal accuracy estimate
44	U4	-	vAcc	mm	Vertical accuracy estimate
48	I4	-	velN	mm/s	NED north velocity
52	I4	-	velE	mm/s	NED east velocity
56	I4	-	velD	mm/s	NED down velocity
60	I4	-	gSpeed	mm/s	Ground Speed (2-D)
64	I4	1e-5	headMot	deg	Heading of motion (2-D)
68	U4	-	sAcc	mm/s	Speed accuracy estimate
72	U4	1e-5	headAcc	deg	Heading accuracy estimate (both motion and vehicle)
76	U2	0.01	pDOP	-	Position DOP
78	X2	-	reserved2	-	Reserved
80	X4	-	reserved3	-	Reserved
84	I4	1e-5	headVeh	deg	Heading of vehicle (2-D)
88	X4	-	reserved4	-	Reserved

Bitfield valid

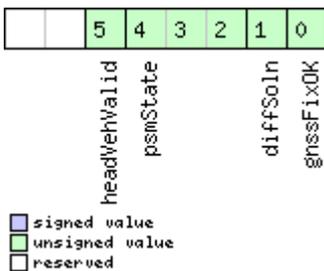
This Graphic explains the bits of valid



Name	Description
validDate	1 = Valid UTC Date
validTime	1 = Valid UTC Time of Day
fullyResolved	1 = UTC Time of Day has been fully resolved (no seconds uncertainty)

Bitfield flags

This Graphic explains the bits of flags



Name	Description
gnssFixOK	A valid fix (i.e within DOP & accuracy masks)
diffSoln	1 if differential corrections were applied
psmState	Power Save Mode state (see Power Management): 0 = n/a (i.e no PSM is active) 1 = ENABLED (an intermediate state before ACQUISITION state) 2 = ACQUISITION 3 = TRACKING 4 = POWER OPTIMIZED TRACKING 5 = INACTIVE
headVehValid	Heading of vehicle is valid

35.10 UBX-NAV-RESETO (0x01 0x10)

35.10.1 Reset odometer

Message	NAV-RESETO					
Description	Reset odometer					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Command					
Comment	This message resets the traveled distance computed by the odometer (see UBX-NAV-ODO). UBX-ACK-ACK or UBX-ACK-NAK are returned to indicate success or failure.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x10	0	see below	CK_A CK_B
No payload						

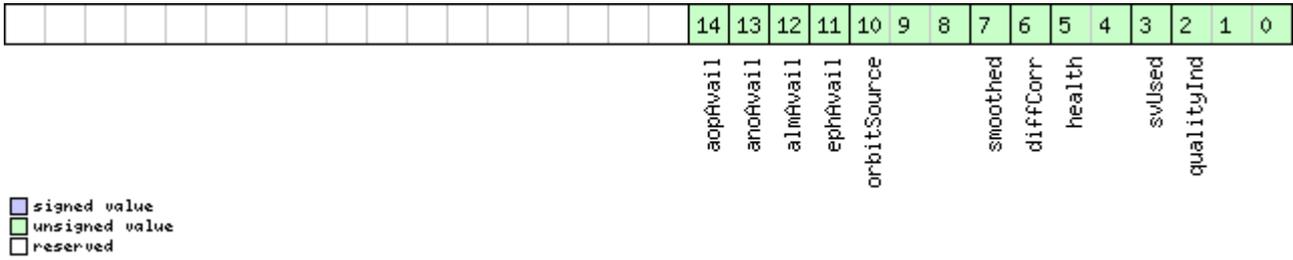
35.11 UBX-NAV-SAT (0x01 0x35)

35.11.1 Satellite Information

Message	NAV-SAT					
Description	Satellite Information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message displays information about SVs which are either known to be visible or currently tracked by the receiver.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x35	8 + 12*numSvs	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U1	-	version	-	Message version (1 for this version)	
5	U1	-	numSvs	-	Number of satellites	
6	U2	-	reserved2	-	Reserved	
Start of repeated block (numSvs times)						
8 + 12*N	U1	-	gnssId	-	GNSS identifier (see Satellite numbering) for assignment	
9 + 12*N	U1	-	svId	-	Satellite identifier (see Satellite numbering) for assignment	
10 + 12*N	U1	-	cno	dBHz	Carrier to noise ratio (signal strength)	
11 + 12*N	I1	-	elev	deg	Elevation (range: +/-90), unknown if out of range	
12 + 12*N	I2	-	azim	deg	Azimuth (range +/-180), unknown if elevation is out of range	
14 + 12*N	I2	0.1	prRes	m	Pseudo range residual	
16 + 12*N	X4	-	flags	-	Bitmask (see graphic below)	
End of repeated block						

Bitfield flags

This Graphic explains the bits of flags



Name	Description
qualityInd	Signal quality indicator: 0: no signal 1: searching signal 2: signal aquired 3: signal detected but unusable 4: code lock on signal 5, 6, 7: code and carrier locked
svUsed	1 = SV is currently being used for navigation
health	SV health flag: 0: unknown 1: healthy 2: unhealthy
diffCorr	1 = differential correction data is available for this SV
smoothed	1 = carrier smoothed pseudorange used
orbitSource	Orbit source: 0: no orbit information is available for this SV 1: ephemeris is used 2: almanac is used 3: AssistNow Offline orbit is used 4: AssistNow Autonomous orbit is used 5, 6, 7: other orbit information is used
ephAvail	1 = ephemeris is available for this SV
almAvail	1 = almanac is available for this SV
anoAvail	1 = AssistNow Offline data is available for this SV
aopAvail	1 = AssistNow Autonomous data is available for this SV

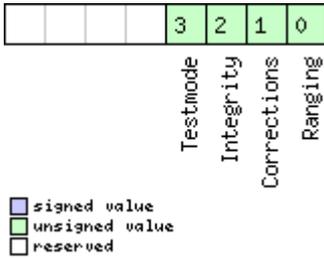
35.12 UBX-NAV-SBAS (0x01 0x32)

35.12.1 SBAS Status Data

Message	NAV-SBAS					
Description	SBAS Status Data					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message outputs the status of the SBAS sub system					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x32	12 + 12*cnt	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U1	-	geo	-	PRN Number of the GEO where correction and integrity data is used from	
5	U1	-	mode	-	SBAS Mode 0 Disabled 1 Enabled Integrity 3 Enabled Testmode	
6	I1	-	sys	-	SBAS System (WAAS/EGNOS/...) -1 Unknown 0 WAAS 1 EGNOS 2 MSAS 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



35.13 UBX-NAV-SOL (0x01 0x06)

35.13.1 Navigation Solution Information

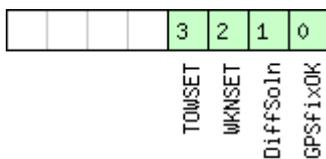
Message	NAV-SOL					
Description	Navigation Solution Information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message combines position, velocity and time solution in ECEF, including accuracy figures. This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-PVT message in preference.					
Message Structure	Header	Class	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 time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	fTOW	ns	Fractional part of iTOW (range: +/-500000). The precise GPS time of week in seconds is: $(iTOW * 1e-3) + (fTOW * 1e-9)$	
8	I2	-	week	weeks	GPS week number of the navigation epoch	
10	U1	-	gpsFix	-	GPSfix Type, range 0..5 0x00 = No Fix 0x01 = Dead Reckoning only 0x02 = 2D-Fix 0x03 = 3D-Fix 0x04 = GPS + dead reckoning combined 0x05 = Time only fix 0x06..0xff: reserved	
11	X1	-	flags	-	Fix Status Flags (see graphic below)	
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	

NAV-SOL continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
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



signed value
 unsigned value
 reserved

Name	Description
GPSfixOK	>1 = Fix within limits (e.g. DOP & accuracy)
DiffSoln	1 = DGPS used
WKNSET	1 = Valid GPS week number
TOWSET	1 = Valid GPS time of week (iTOW & fTOW)

35.14 UBX-NAV-STATUS (0x01 0x03)
35.14.1 Receiver Navigation Status

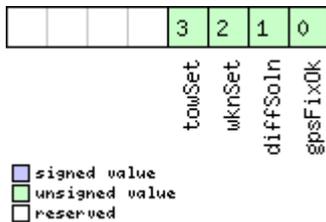
Message	NAV-STATUS					
Description	Receiver Navigation Status					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	See important comments concerning validity of position and velocity given in section Navigation Output Filters. -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x03	16	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	

NAV-STATUS continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
4	U1	-	gpsFix	-	GPSfix Type, this value does not qualify a fix as valid and within the limits. See note on flag gpsFixOk below. 0x00 = no fix 0x01 = dead reckoning only 0x02 = 2D-fix 0x03 = 3D-fix 0x04 = GPS + dead reckoning combined 0x05 = Time only fix 0x06..0xff = reserved
5	X1	-	flags	-	Navigation Status Flags (see graphic below)
6	X1	-	fixStat	-	Fix Status Information (see graphic below)
7	X1	-	flags2	-	further information about navigation output (see graphic below)
8	U4	-	ttff	-	Time to first fix (millisecond time tag)
12	U4	-	msss	-	Milliseconds since Startup / Reset

Bitfield flags

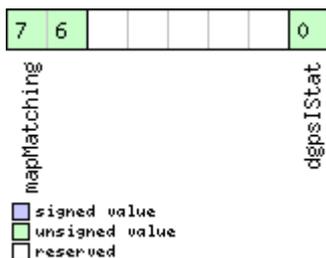
This Graphic explains the bits of flags



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



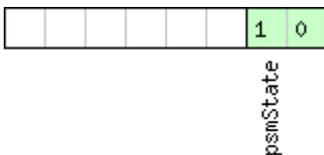
Name	Description
------	-------------

Bitfield fixStat Description continued

Name	Description
dgpsIStat	DGPS Input Status 0: none 1: PR+PRR Correction
mapMatching	map matching status: 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 <code>gpsFix</code> type instead of publishing no fix

Bitfield flags2

This Graphic explains the bits of `flags2`



signed value
 unsigned value
 reserved

Name	Description
psmState	power save mode state 0: ACQUISITION [or when psm disabled] 1: TRACKING 2: POWER OPTIMIZED TRACKING 3: INACTIVE

35.15 UBX-NAV-SVINFO (0x01 0x30)

35.15.1 Space Vehicle Information

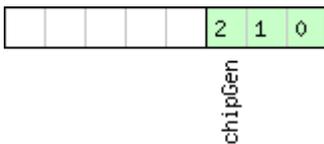
Message	NAV-SVINFO					
Description	Space Vehicle Information					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	Information about satellites used or visible This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-SAT message in preference.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x30	8 + 12*numCh	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	U1	-	numCh	-	Number of channels	
5	X1	-	globalFlags	-	Bitmask (see graphic below)	
6	U2	-	reserved2	-	Reserved	

NAV-SVINFO continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
<i>Start of repeated block (numCh times)</i>					
8 + 12*N	U1	-	chn	-	Channel number, 255 for SVs not assigned to a channel
9 + 12*N	U1	-	svid	-	Satellite ID, see Satellite numbering for assignment
10 + 12*N	X1	-	flags	-	Bitmask (see graphic below)
11 + 12*N	X1	-	quality	-	Bitfield (see graphic below)
12 + 12*N	U1	-	cno	dBHz	Carrier to Noise Ratio (Signal Strength)
13 + 12*N	I1	-	elev	deg	Elevation in integer degrees
14 + 12*N	I2	-	azim	deg	Azimuth in integer degrees
16 + 12*N	I4	-	prRes	cm	Pseudo range residual in centimetres
<i>End of repeated block</i>					

Bitfield globalFlags

This Graphic explains the bits of globalFlags

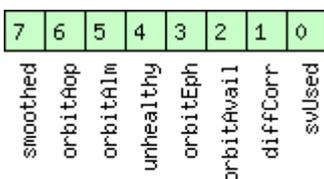


signed value
 unsigned value
 reserved

Name	Description
chipGen	Chip hardware generation 0: Antaris, Antaris 4 1: u-blox 5 2: u-blox 6 3: u-blox 7 4: u-blox M8

Bitfield flags

This Graphic explains the bits of flags



signed value
 unsigned value
 reserved

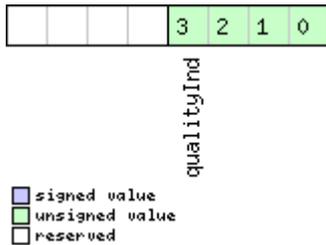
Name	Description
svUsed	SV is used for navigation
diffCorr	Differential correction data is available for this SV
orbitAvail	Orbit information is available for this SV (Ephemeris or Almanac)
orbitEph	Orbit information is Ephemeris
unhealthy	SV is unhealthy / shall not be used

Bitfield flags Description continued

Name	Description
orbitAlm	Orbit information is Almanac Plus
orbitAop	Orbit information is AssistNow Autonomous
smoothed	Carrier smoothed pseudorange used

Bitfield quality

This Graphic explains the bits of quality



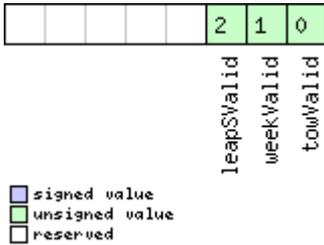
Name	Description
qualityInd	Signal Quality indicator (range 0..7). The following list shows the meaning of the different QI values: 0: This channel is idle 1: Channel is searching 2: Signal aquired 3: Signal detected but unusable 4: Code Lock on Signal 5, 6, 7: Code and Carrier locked

35.16 UBX-NAV-TIMEGPS (0x01 0x20)
35.16.1 GPS Time Solution

Message	NAV-TIMEGPS					
Description	GPS Time Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message reports the precise GPS time of the most recent navigation solution including validity falgs and an accuracy estimate.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x20	16	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	fTOW	ns	Fractional part of iTOW (range: +/-500000). The precise GPS time of week in seconds is: (iTOW * 1e-3) + (fTOW * 1e-9)	
8	I2	-	week	-	GPS week number of the navigation epoch	
10	I1	-	leapS	s	GPS leap seconds (GPS-UTC)	
11	X1	-	valid	-	Validity Flags (see graphic below)	
12	U4	-	tAcc	ns	Time Accuracy Estimate	

Bitfield valid

This Graphic explains the bits of valid



Name	Description
towValid	1 = Valid GPS time of week (iTOW & fTOW)
weekValid	1 = Valid GPS week number
leapSValid	1 = Valid GPS leap seconds

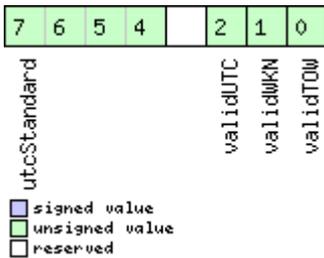
35.17 UBX-NAV-TIMEUTC (0x01 0x21)

35.17.1 UTC Time Solution

Message	NAV-TIMEUTC					
Description	UTC Time Solution					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	Note that during a leap second there may be more or less than 60 seconds in a minute; see the description of leap seconds for details. -					
Message Structure	Header	Class	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 time of week of the navigation epoch . See the description of iTOW for details.	
4	U4	-	tAcc	ns	Time accuracy estimate (UTC)	
8	I4	-	nano	ns	Fraction of second, range -1e9 .. 1e9 (UTC)	
12	U2	-	year	y	Year, range 1999..2099 (UTC)	
14	U1	-	month	month	Month, range 1..12 (UTC)	
15	U1	-	day	d	Day of month, range 1..31 (UTC)	
16	U1	-	hour	h	Hour of day, range 0..23 (UTC)	
17	U1	-	min	min	Minute of hour, range 0..59 (UTC)	
18	U1	-	sec	s	Seconds of minute, range 0..60 (UTC)	
19	X1	-	valid	-	Validity Flags (see graphic below)	

Bitfield valid

This Graphic explains the bits of valid



Name	Description
validTOW	1 = Valid Time of Week
validWKN	1 = Valid Week Number
validUTC	1 = Valid UTC Time
utcStandard	UTC standard identifier. 0: Information not available 1: Communications Research Laboratory (CRL) 2: National Institute of Standards and Technology (NIST) 3: U.S. Naval Observatory (USNO) 4: International Bureau of Weights and Measures (BIPM) 5: European Laboratory (tbd) 6: Former Soviet Union (SU) 15: Unknown

35.18 UBX-NAV-VELECEF (0x01 0x11)

35.18.1 Velocity Solution in ECEF

Message	NAV-VELECEF					
Description	Velocity Solution in ECEF					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	See important comments concerning validity of velocity given in section Navigation Output Filters. -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x11	20	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	ecefVX	cm/s	ECEF X velocity	
8	I4	-	ecefVY	cm/s	ECEF Y velocity	
12	I4	-	ecefVZ	cm/s	ECEF Z velocity	
16	U4	-	sAcc	cm/s	Speed accuracy estimate	

35.19 UBX-NAV-VELNED (0x01 0x12)

35.19.1 Velocity Solution in NED

Message	NAV-VELNED					
Description	Velocity Solution in NED					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	See important comments concerning validity of velocity given in section Navigation Output Filters. -					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x12	36	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.	
4	I4	-	velN	cm/s	North velocity component	
8	I4	-	velE	cm/s	East velocity component	
12	I4	-	velD	cm/s	Down velocity component	
16	U4	-	speed	cm/s	Speed (3-D)	
20	U4	-	gSpeed	cm/s	Ground speed (2-D)	
24	I4	1e-5	heading	deg	Heading of motion 2-D	
28	U4	-	sAcc	cm/s	Speed accuracy Estimate	
32	U4	1e-5	cAcc	deg	Course / Heading accuracy estimate	

36 UBX-RXM (0x02)

Receiver Manager Messages: i.e. Satellite Status, RTC Status.

Messages in Class RXM output status and result data from the Receiver Manager.

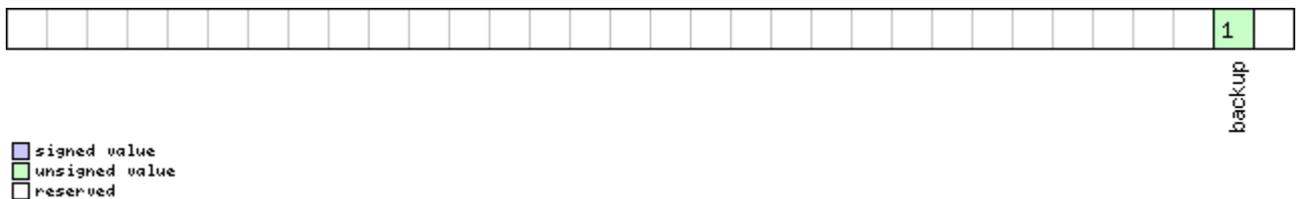
36.1 UBX-RXM-PMREQ (0x02 0x41)

36.1.1 Requests a Power Management task

Message	RXM-PMREQ					
Description	Requests a Power Management task					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Command					
Comment	Request of a Power Management related task of the receiver.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x02	0x41	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

36.2 UBX-RXM-SVSI (0x02 0x20)

36.2.1 SV Status Info

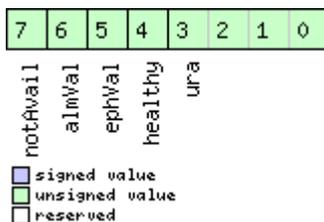
Message	RXM-SVSI					
Description	SV Status Info					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	Status of the receiver manager knowledge about GPS Orbit Validity This message has only been retained for backwards compatibility; users are recommended to use the UBX-NAV-ORB message in preference.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x02	0x20	8 + 6*numSV	see below	CK_A CK_B
Payload Contents:						

RXM-SVSI continued

Byte Offset	Number Format	Scaling	Name	Unit	Description
0	U4	-	iTOW	ms	GPS time of week of the navigation epoch . See the description of iTOW for details.
4	I2	-	week	weeks	GPS week number of the navigation epoch
6	U1	-	numVis	-	Number of visible satellites
7	U1	-	numSV	-	Number of per-SV data blocks following
<i>Start of repeated block (numSV times)</i>					
8 + 6*N	U1	-	svid	-	Satellite ID
9 + 6*N	X1	-	svFlag	-	Information Flags (see graphic below)
10 + 6*N	I2	-	azim	-	Azimuth
12 + 6*N	I1	-	elev	-	Elevation
13 + 6*N	X1	-	age	-	Age of Almanac and Ephemeris: (see graphic below)
<i>End of repeated block</i>					

Bitfield svFlag

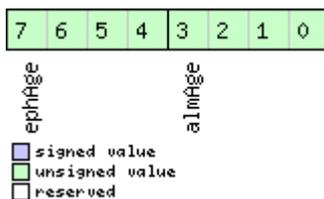
This Graphic explains the bits of svFlag



Name	Description
ura	Figure of Merit (URA) range 0..15
healthy	SV healthy flag
ephVal	Ephemeris valid
almVal	Almanac valid
notAvail	SV not available

Bitfield age

This Graphic explains the bits of age



Name	Description
almAge	Age of ALM in days offset by 4 i.e. the reference time may be in the future: ageOfAlm = (age & 0x0f) - 4

Bitfield age Description continued

<i>Name</i>	<i>Description</i>
ephAge	Age of EPH in hours offset by 4. i.e. the reference time may be in the future: $\text{ageOfEph} = (\text{age} \& 0xf0) \gg 4 - 4$

37 UBX-TIM (0x0D)

Timing Messages: i.e. Time Pulse Output, Timemark Results.

Messages in this class are output by the receiver, giving information on Timepulse and Timemark measurements.

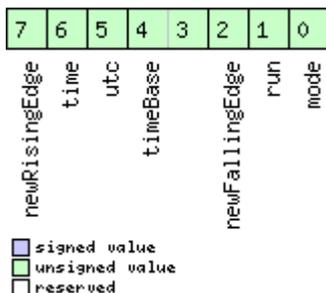
37.1 UBX-TIM-TM2 (0x0D 0x03)

37.1.1 Time mark data

Message	TIM-TM2					
Description	Time mark data					
Firmware	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message contains information for high precision time stamping / pulse counting. The delay figures and timebase given in CFG-TP5 are also applied to the time results output in this message.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x0D	0x03	28	see below	CK_A CK_B
Payload Contents:						
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

Bitfield flags Description continued

Name	Description
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

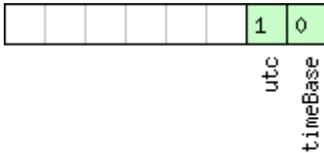
37.2 UBX-TIM-TP (0x0D 0x01)

37.2.1 Time Pulse Timedata

Message	TIM-TP					
Description	Time Pulse Timedata					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Periodic/Polled					
Comment	This message contains information for high precision timing. The recommended configuration when using this message is to set both the measurement rate (CFG-RATE) and the timepulse frequency (CFG-TP5) to 1Hz. For more information see section Time pulse .					
Message Structure	Header	Class	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	Time pulse time of week according to time base	
4	U4	2 ⁻³²	towSubMS	ms	Submillisecond part of TOWMS	
8	I4	-	qErr	ps	Quantization error of time pulse.	
12	U2	-	week	weeks	Time pulse 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



signed value
 unsigned value
 reserved

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

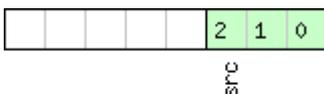
37.3 UBX-TIM-VERFY (0x0D 0x06)

37.3.1 Sourced Time Verification

Message	TIM-VERFY					
Description	Sourced Time Verification					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Polled/Once					
Comment	This message contains verification information about previous time received via AID-INI or from RTC					
Message Structure	Header	Class	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



signed value
 unsigned value
 reserved

Name	Description
------	-------------

Bitfield flags Description continued

<i>Name</i>	<i>Description</i>
src	aiding time source 0: no time aiding done 2: source was RTC 3: source was AID-INT

38 UBX-UPD (0x09)

Firmware Update Messages: i.e. Memory/Flash erase/write, Reboot, Flash identification, etc..

Messages in this class are used to update the firmware.

38.1 UBX-UPD-SOS (0x09 0x14)

38.1.1 Poll Backup File Restore Status

Message	UPD-SOS					
Description	Poll Backup File Restore Status					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Poll Request					
Comment	Sending this (empty / no-payload) message to the receiver results in the receiver returning a <i>System Restored from Backup</i> message as defined below.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x09	0x14	0	see below	CK_A CK_B
No payload						

38.1.2 Create Backup File in Flash

Message	UPD-SOS					
Description	Create Backup File in Flash					
Firmware	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
Type	Input					
Comment	The host can send this message in order to save part of the BBR memory in a file in flash file system. The feature is designed in order to emulate the presence of the backup battery even if it is not present; the host can issue the save on shutdown command before switching off the device supply. It is recommended to issue a GNSS stop command before, in order to keep the BBR memory content consistent.					
Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x09	0x14	4	see below	CK_A CK_B
Payload Contents:						
Byte Offset	Number Format	Scaling	Name	Unit	Description	
0	U1	-	cmd	-	Command (must be 0)	
1	U1[3]	-	reserved0	-	reserved (must be 0)	

38.1.3 Clear Backup in Flash

<i>Message</i>	UPD-SOS					
<i>Description</i>	Clear Backup in Flash					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Input					
<i>Comment</i>	The host can send this message in order to erase the backup file present in flash. It is recommended that the clear operation is issued after the host has received the notification that the memory has been restored after a reset. Alternatively the host can parse the startup string 'Restored data saved on shutdown' or poll the UBX-UPD-SOS message for getting the status.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x09	0x14	4	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	cmd	-	Command (must be 1)	
1	U1[3]	-	reserved0	-	reserved (must be 0)	

38.1.4 Backup File Creation Acknowledge

<i>Message</i>	UPD-SOS					
<i>Description</i>	Backup File Creation Acknowledge					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> u-blox M8 firmware version 2.00 					
<i>Type</i>	Output					
<i>Comment</i>	The message is sent from the device as confirmation of creation of a backup file in flash. The host can safely shut down the device after received this message.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x09	0x14	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	cmd	-	Command (must be 2)	
1	U1[3]	-	reserved0	-	reserved (must be 0)	
4	U1	-	response	-	0: Not acknowledged 1: Acknowledged	
5	U1[3]	-	reserved1	-	reserved (must be 0)	

38.1.5 System Restored from Backup

<i>Message</i>	UPD-SOS					
<i>Description</i>	System Restored from Backup					
<i>Firmware</i>	Supported on: <ul style="list-style-type: none"> • u-blox M8 firmware version 2.00 					
<i>Type</i>	Output					
<i>Comment</i>	The message is sent from the device to notify the host the BBR has been restored from a backup file in flash. The host should clear the backup file after receiving this message. If the UBX-UPD-SOS message is polled, this message will be resent.					
<i>Message Structure</i>	<i>Header</i>	<i>Class</i>	<i>ID</i>	<i>Length (Bytes)</i>	<i>Payload</i>	<i>Checksum</i>
	0xB5 0x62	0x09	0x14	8	see below	CK_A CK_B
<i>Payload Contents:</i>						
<i>Byte Offset</i>	<i>Number Format</i>	<i>Scaling</i>	<i>Name</i>	<i>Unit</i>	<i>Description</i>	
0	U1	-	cmd	-	Command (must be 3)	
1	U1[3]	-	reserved0	-	reserved (must be 0)	
4	U1	-	response	-	0: Unknown 1: Failed restoring from backup file 2: Restored from backup file 3: Not restored (no backup)	
5	U1[3]	-	reserved1	-	reserved (must be 0)	

RTCM Protocol

39 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.rtcn.org>.



This feature is only applicable to GPS operation.



For effective differential positioning accuracy, it is necessary that the reference station antenna is situated in a low multipath environment with an unobstructed view of the sky. It is recommended that reference receiver applies phase smoothing to the broadcast corrections.

40 Supported Messages

The following RTCM 2.3 messages are supported:

Supported RTCM 2.3 Message Types

Message Type	Description
1	Differential GPS Corrections
2	Delta Differential GPS Corrections
3	GPS Reference Station Parameters
9	GPS Partial Correction Set

41 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. However, the RTCM protocol must be enabled on the interface used by means of the [UBX-CFG-PRT](#) message.

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.

42 Output

DGPS mode will result in following modified output:

- [NMEA-GGA](#): The quality field will be 2 (see [NMEA Positon Fix Flags](#)). The age of DGPS corrections and Reference station ID will be set.
- [NMEA-GLL](#), [NMEA-RMC](#), [NMEA-VTG](#), [NMEA-GNS](#): The posMode indicator will be D (see [NMEA Positon Fix Flags](#)).
- [NMEA-PUBX-POSITION](#): The status will be D2/D3; The age of DGPS corrections will be set.
- [UBX-NAV-SOL](#): The DGPS will be set.
- [UBX-NAV-PVT](#): 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"

43 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*.

44 Reference

The RTCM support is implemented according to RTCM 10402.3 ("RECOMMENDED STANDARDS FOR DIFFERENTIAL GNSS").

Appendix

A Protocol Versions

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

Each receiver reports its supported Protocol Version in the following ways:

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

The following tables show the supported Protocol Versions for a number of common firmware versions and platforms.

A.1 Supported Protocol Versions

u-blox 5

<i>Firmware Version</i>	<i>Supported Protocol Version</i>
4.00	10.00
4.01	10.01
5.00	11.00
6.00	12.00
6.02	12.02

u-blox 6

<i>Firmware Version</i>	<i>Supported Protocol Version</i>
6.00	12.00
6.02	12.02
7.01	13.01
7.03	13.03

u-blox 6 GPS/GLONASS/QZSS

<i>Firmware Version</i>	<i>Supported Protocol Version</i>
1.00	14.00

u-blox 7

<i>Firmware Version</i>	<i>Supported Protocol Version</i>
1.00	14.00
1.01	14.00

u-blox M8

<i>Firmware Version</i>	<i>Supported Protocol Version</i>
2.00	15.00

B Satellite Numbering

A summary of all the SV numbering schemes is provided in the following table.

Satellite numbering

<i>GNSS Type</i>	<i>SV range</i>	<i>UBX gnssId:svId</i>	<i>UBX svId</i>	<i>NMEA 2.X (strict)</i>	<i>NMEA 2.X (extended)</i>	<i>NMEA 4.X (strict)</i>	<i>NMEA 4.X (extended)</i>

Satellite numbering continued

GNSS Type	SV range	UBX gnssId:svId	UBX svId	NMEA 2.X (strict)	NMEA 2.X (extended)	NMEA 4.X (strict)	NMEA 4.X (extended)
GPS	G1-G32	0:1-32	1-32	1-32	1-32	1-32	1-32
SBAS	S120-S158	1:120-158	120-158	33-64	33-64,152-158	33-64	33-64,152-158
Galileo	E1-E36	2:1-36	211-246	-	301-336	1-36	1-36
BeiDou	B1-B37	3:1-37	159-163,33-64	-	401-437	1-37	1-37
QZSS	Q1-Q5	5:1-5	193-197	-	193-197	-	193-197
GLONASS	R1-R32, R?	6:1-32, 6:255	65-96, 255	65-96, null	65-96, null	65-96, null	65-96, null

C u-blox M8 Default Settings

The default settings listed in this section apply from u-blox M8 ROM-based receivers with ROM version 2.00 and above. These values assume that the default levels of the configuration pins have been left unchanged and no setting that affects the default configuration was written to the eFuse. Default settings are dependent on the configuration pin and eFuse settings, for information regarding these settings, consult the applicable Data Sheet.

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

For parameter and protocol description see section [UBX-CFG-ANT](#).

Antenna Settings

Parameter	Description	Default Setting	Unit
flags-svcs	Enable Control Signal	Enabled	
flags-scd	Enable Short Circuit Detection	Enabled	
flags-pdwnOnSCD	Enable Short Circuit Power Down logic	Enabled	
flags-recovery	Enable Automatic Short Circuit Recovery logic	Enabled	
flags-ocd	Enable Open Circuit Detection	Disabled	
pins-pinSwitch	PIO-Pin used for switching antenna supply	16	
pins-pinSCD	PIO-Pin used for detecting a short in the antenna supply	15	
pins-pinOCD	PIO-Pin used for detecting open/not connected antenna	14	

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

For parameter and protocol description see section [UBX-CFG-DAT](#).

Datum Default Settings

Parameter	Description	Default Setting	Unit
datumNum	Datum number	0	
datumName	Datum name	WGS84	
majA	Semi-major Axis	6378137	m
flat	1.0 / Flattening	298.257223563	
dX	X Axis shift at the origin	0	m
dY	Y Axis shift at the origin	0	m
dZ	Z Axis shift at the origin	0	m
rotX	Rotation about the X Axis	0	s
rotY	Rotation about the Y Axis	0	s
rotZ	Rotation about the Z Axis	0	s
scale	Scale change	0	ppm

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

For parameter and protocol description see section [UBX-CFG-NAV5](#).

Navigation Default Settings

Parameter	Description	Default Setting	Unit
dynModel	Dynamic Platform Model	0 - Portable	
fixMode	Fix Mode	3 - Auto 2D/3D	
fixedAlt	Fixed Altitude	N/A (fixMode=3)	m
fixedAltVar	Fixed Altitude Variance	N/A (fixMode=3)	m ²
minElev	Min SV Elevation	5	deg
pDop	PDOP Mask	25	-
tDop	TDOP Mask	25	-
pAcc	P Accuracy	100	m
tAcc	T Accuracy	300	m
staticHoldThresh	Static Hold Threshold	0.00	cm/s
dgpsTimeOut	DGPS timeout	60	s
cnoThreshNumSVs	Number of SVs required to have C/N0 above cnoThresh for a valid fix	0	
cnoThresh	C/N0 threshold for a valid fix	0	dBHz
pAccAdr	ADR Position Accuracy Mask	0	m
staticHoldMaxDist	Static hold distance threshold	0	m/s



The Dynamic Platform Model default setting is different for certain product variants.

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

For parameter and protocol description see section [UBX-CFG-NAVX5](#).

Navigation Default Settings

Parameter	Description	Default Setting	Unit
minSVs	Minimum number of SV	3	
maxSVs	Maximum number of SV	20	
minCNO	Minimum C/N0 for navigation	6	dBHz
iniFix3D	Initial Fix must be 3D	Disabled	
wknRollover	Weeknumber rollover	1756	
usePPP	Use PPP	disabled	
aopCfg-useAOP	Use AssistNow Autonomous	Disabled	
aopOrbMaxErr	AssistNow Autonomous max. acceptable orbit error	100	m



The minimum number of SV default setting is different for certain product variants.

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

For parameter and protocol description see section [UBX-CFG-RATE](#).

Output Rate Default Settings

Parameter	Description	Default Setting	Unit
timeRef	Time Source	1 – GPS time	
measRate	Measurement Period	1000	ms
navRate	Measurement Rate	1	cycles

C.6 Power Management 2 Configuration (UBX-CFG-PM2)

For parameter and protocol description see section [UBX-CFG-PM2](#).

Power Management 2 Configuration Default Settings

Parameter	Description	Default Setting	Unit
version	Version	1	
flags-extintSelect	EXTINT pin selection	EXTINT0	
flags-extintWake	EXTINT pin control - keep awake	Disabled	
flags-extintBackup	EXTINT pin control - force backup	Disabled	
flags-limitPeakCurr	Limit peak current	Disabled	
flags-WaitTimeFix	Wait for time fix	Disabled	
flags-updateRTC	Update Real Time Clock	Disabled	
flags-updateEPH	Update ephemeris	Enabled	
flags-doNotEnterOff	Do not enter 'inactive for search' state when no fix	Disabled	
flags-mode	Mode of operation	Cyclic tracking	
updatePeriod	Update period	1000	ms
searchPeriod	Search period	10000	ms
gridOffset	Grid offset	0	ms
onTime	On time	0	s
minAcqTime	Minimum acquisition time	0	s

C.7 Receiver Manager Configuration (UBX-CFG-RXM)

For parameter and protocol description see section [UBX-CFG-RXM](#).

Power Management Default Settings

Parameter	Description	Default Setting	Unit
lpMode	Low power mode	0 - Continuous Mode	

C.8 GNSS system configuration (UBX-CFG-GNSS)

For parameter and protocol description see section [UBX-CFG-GNSS](#).

UBX-CFG-GNSS Default Settings

Parameter	Description	Default Setting	Unit
numTrkChHw	Number of available tracking channels	32	
numTrkChUse	Number of tracking channels to use	32	
numConfigBlocks	Number of configuration blocks following	5	
gnssId	GNSS identifier (see Satellite Numbering)	0, 1, 3, 5, 6	
flags-enable	Enable this GNSS system (see Satellite Numbering)	0, 1, 5, 6	
resTrkCh	Minimum number of tracking channels per GNSS	8, 1, 0, 8	
maxTrkCh	Maximum number of tracking channels per GNSS	16, 3, 3, 14	

C.9 SBAS Configuration (UBX-CFG-SBAS)

For parameter and protocol description see section [UBX-CFG-SBAS](#).

SBAS Configuration Default Settings

Parameter	Description	Default Setting	Unit
mode-enabled	SBAS Subsystem	Enabled	
mode-test	Allow test mode usage	Disabled	

SBAS Configuration Default Settings continued

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
usage-range	Ranging (Use SBAS for navigation)	Enabled	
usage-diffCorr	Apply SBAS Correction Data	Enabled	
usage-integrity	Apply integrity information	Disabled	
maxSBAS	Maximum number of SBAS tracking channels	3	
scanmode1	PRN Codes 120-151	120, 124, 126, 129, 133, 135, 137, 138	
scanmode2	PRN Codes 152-158	None	

C.10 Port Configuration (UBX-CFG-PRT)

For parameter and protocol description see section [UBX-CFG-PRT](#).

C.10.1 UART Port Configuration

For parameter and protocol description see section [UBX-CFG-PRT-UART](#).

UART 1 Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
portID	Port ID	1 (UART 1)	
txReady-en	TX-ready feature	0 (disabled)	
mode-charLen	Character Length	3 (8 bit)	
mode-parity	Parity	4 (No parity)	
mode-nStopBits	Number of Stop Bits	0 (1 stop bit)	
baudRate	Baud rate	9600	baud
inProtoMask	Protocol in	inUBX, inNMEA, inRTCM	
outProtoMask	Protocol out	outUBX, outNMEA	
flags-extendedTxTimeout	Extended TX timeout	0 - disabled	

C.10.2 USB Port Configuration

For parameter and protocol description see section [UBX-CFG-PRT-USB](#).

USB Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
portID	Port ID	3 (USB)	
txReady-en	TX-ready feature	0 (disabled)	
inProtoMask	Protocol in	inUBX, inNMEA, inRTCM	
outProtoMask	Protocol out	outUBX, outNMEA	

C.10.3 SPI Port Configuration

For parameter and protocol description see section [UBX-CFG-PRT-SPI](#).

SPI Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
portID	Port ID	4 (SPI)	
txReady-en	TX-ready feature	0 (disabled)	
mode-spiMode	SPI mode	0 (CPOL=0, CPHA=0)	
mode-ffCnt	0xFF count	50	
inProtoMask	Protocol in	inUBX, inNMEA, inRTCM	
outProtoMask	Protocol out	outUBX, outNMEA	

SPI Default Settings continued

Parameter	Description	Default Setting	Unit
flags-extendedTxTimeout	Extended TX timeout	0 - disabled	

C.10.4 DDC Port Configuration

For parameter and protocol description see section [UBX-CFG-PRT-DDC](#).

DDC Default Settings

Parameter	Description	Default Setting	Unit
portID	Port ID	0 (DDC)	
txReady-en	TX-ready feature	0 (disabled)	
mode-slaveAddr	Slave address	0x42	
inProtoMask	Protocol in	inUBX, inNMEA, inRTCM	
outProtoMask	Protocol out	outUBX, outNMEA	
flags-extendedTxTimeout	Extended TX timeout	0 - disabled	

C.11 USB Settings (UBX-CFG-USB)

For parameter and protocol description see section [UBX-CFG-USB](#).

USB default settings

Parameter	Description	Default Setting	Unit
vendorID	Vendor ID	0x1546	
productID	Product ID	0x01A8	
powerConsumption	Bus Current required	100	mA
flags-powerMode	Power Mode	1 (self-powered)	
vendorString	Vendor string	u-blox AG - www. u-blox.com	
productString	Product string	u-blox GNSS receiver	
serialNumber	Serial number		

C.12 Message Settings (UBX-CFG-MSG)

For parameter and protocol description see section [UBX-CFG-MSG](#).

Enabled output messages

Message	Type	All Ports
NMEA-Standard-GGA	Out	1
NMEA-Standard-GLL	Out	1
NMEA-Standard-GSA	Out	1
NMEA-Standard-GSV	Out	1
NMEA-Standard-RMC	Out	1
NMEA-Standard-VTG	Out	1

C.13 NMEA Protocol Settings (UBX-CFG-NMEA)

For parameter and protocol description see section [UBX-CFG-NMEA](#).

NMEA Protocol Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
filter-posFilt	Enable position output even for failed or invalid fixes	Disabled	
filter-mskPosFilt	Enable position even for invalid fixes	Disabled	
filter-timeFilt	Enable time output even for invalid times	Disabled	
filter-dateFilt	Enable time output even for invalid dates	Disabled	
filter-gpsOnlyFilter	Restrict output to GPS satellites only	Disabled	
filter-trackFilt	Enable COG output even if COG is frozen	Disabled	
nmeaVersion	NMEA version	4.0	
numSV	Number of SVs to report	Unlimited	
flags-compat	Compatibility Mode	Disabled	
flags-consider	Consideration Mode	Enabled	
gnssToFilter-gps	Disable GPS satellites	False	
gnssToFilter-sbas	Disable SBAS satellites	False	
gnssToFilter-qzss	Disable QZSS satellites	False	
gnssToFilter-glonass	Disable GLONASS satellites	False	
gnssToFilter-beidou	Disable BeiDou satellites	False	
svNumbering	Output of SV's with no NMEA defined value	0 (not output)	
mainTalkerId	Override main Talker ID	0 (not overridden)	
gsvTalkerId	Override GSV Talker ID	0 (not overridden)	
bdsTalkerId	Set BeiDou Talker ID (two characters)	0 (not overridden)	

C.14 Logging Configuration (UBX-CFG-LOGFILTER)

For parameter and protocol description see section [UBX-CFG-LOGFILTER](#).

UBX-CFG-LOGFILTER Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
flags-recordEnabled	Recording enabled	0	
flags-applyAllFilterSettings	Apply all filter settings	0	
flags-psmOncePerWakeupEnabled	Recording of single position per PSM wake up enabled	0	
minInterval	Minimum time interval	0	s
timeThreshold	Time threshold	0	s
speedThreshold	Speed threshold	0	m/s
positionThreshold	Position threshold	0	m

C.15 Remote Inventory (UBX-CFG-RINV)

For parameter and protocol description see section [UBX-CFG-RINV](#).

UBX-CFG-RINV Default Settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
flags-dump	Dump data at startup	0	
flags-binary	Data is binary	0	
data	Data stored in Remote Inventory	Notice: no data saved!	

C.16 INF Messages Settings (UBX-CFG-INF)

For parameter and protocol description see section [UBX-CFG-INF](#).

INF messages default settings

Parameter	Type	All Ports	Range/Remark
infMsgMask-ERROR	Out	1	In NMEA Protocol only (GPTXT)
infMsgMask-WARNING	Out	1	In NMEA Protocol only (GPTXT)
infMsgMask-NOTICE	Out	1	In NMEA Protocol only (GPTXT)
infMsgMask-TEST	Out		
infMsgMask-DEBUG	Out		

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

For parameter and protocol description see section [UBX-CFG-TP5](#).

TIMEPULSE default settings

Parameter	Description	Default Setting	Unit
tpIdx	Time pulse selection	0	
antCableDelay	Cable Delay	50	ns
rfGroupDelay	RF Groupdelay	0	ns
freqPeriod	Period	1000000	us
freqPeriodLock	Period Locked	1000000	us
pulseLenRatio	Pulse Length	0	us
pulseLenRatioLock	Pulse Length Locked	100000	us
userConfigDelay	User Delay	0	ns
flags-gridUtcGps	Timegrid	0 (UTC Time)	
flags-polarity	Polarity	1 (rising edge at top of second)	
flags-alignToTow	Align to TOW	1	
flags-isLength	IsLength	1	
flags-isFreq	IsFreq	0	
flags-lockedOtherSet	Locked other setting	1	
flags-lockGnssFreq	Lock to GNSS freq	1	
flags-Active	Active	1	

TIMEPULSE2 default settings

Parameter	Description	Default Setting	Unit
tpIdx	Time pulse selection	1	
antCableDelay	Cable Delay	50	ns
rfGroupDelay	RF Groupdelay	0	ns
freqPeriod	Frequency	4	Hz
freqPeriodLock	Frequency Locked	1	Hz
pulseLenRatio	Pulse Length	125000	us
pulseLenRatioLock	Pulse Length Locked	100000	us
userConfigDelay	User Delay	0	ns
flags-gridUtcGps	Timegrid	0 (UTC Time)	
flags-polarity	Polarity	1 (rising edge at top of second)	
flags-alignToTow	Align to TOW	1	
flags-isLength	IsLength	1	
flags-isFreq	IsFreq	1	
flags-lockedOtherSet	Locked other setting	1	

TIMEPULSE2 default settings continued

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
flags-lockGnssFreq	Lock to GNSS freq	1	
flags-Active	Active	0	

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

For parameter and protocol description see section [UBX-CFG-ITFM](#).

Jamming/Interference monitor default settings

<i>Parameter</i>	<i>Description</i>	<i>Default Setting</i>	<i>Unit</i>
config-enable	Enable	Disabled	
config-bbThreshold	Broadband interference detection threshold	3	dB
config-cwThreshold	CW interference detection threshold	15	dB
config-antSetting	Antenna setting	0	

D u-blox M8 Standard firmware versions

Standard FW version strings

<i>Generation</i>	<i>Version</i>	<i>String</i>	<i>ROM BASE</i>
u-blox M8	FW 2.00	ROM CORE 2.00 (74168) Sep 26 2013 12:08:50	-
u-blox M8	FW 2.00	EXT CORE 2.00 (74182) Sep 26 2013 14:42:35	u-blox M8 ROM 0.22 - 2.00

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.

Revision History

<i>Revision</i>	<i>Date</i>	<i>Name</i>	<i>Status / Comments</i>
R01	30 Sep 2013	efav	adding u-blox M8 firmware 2.00



Contact

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

u-blox Offices

North, Central and South America

u-blox America, Inc.

Phone: +1 703 483 3180
E-mail: info_us@u-blox.com

Regional Office West Coast:

Phone: +1 408 573 3640
E-mail: info_us@u-blox.com

Technical Support:

Phone: +1 703 483 3185
E-mail: support_us@u-blox.com

Headquarters

Europe, Middle East, Africa

u-blox AG

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

Asia, Australia, Pacific

u-blox Singapore Pte. Ltd.

Phone: +65 6734 3811
E-mail: info_ap@u-blox.com
Support: support_ap@u-blox.com

Regional Office Australia:

Phone: +61 2 8448 2016
E-mail: info_au@u-blox.com
Support: support_ap@u-blox.com

Regional Office China (Beijing):

Phone: +86 10 68 133 545
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office China (Shenzhen):

Phone: +86 755 8627 1083
E-mail: info_cn@u-blox.com
Support: support_cn@u-blox.com

Regional Office India:

Phone: +91 959 1302 450
E-mail: info_in@u-blox.com
Support: support_in@u-blox.com

Regional Office Japan:

Phone: +81 3 5775 3850
E-mail: info_jp@u-blox.com
Support: support_jp@u-blox.com

Regional Office Korea:

Phone: +82 2 542 0861
E-mail: info_kr@u-blox.com
Support: support_kr@u-blox.com

Regional Office Taiwan:

Phone: +886 2 2657 1090
E-mail: info_tw@u-blox.com
Support: support_tw@u-blox.com