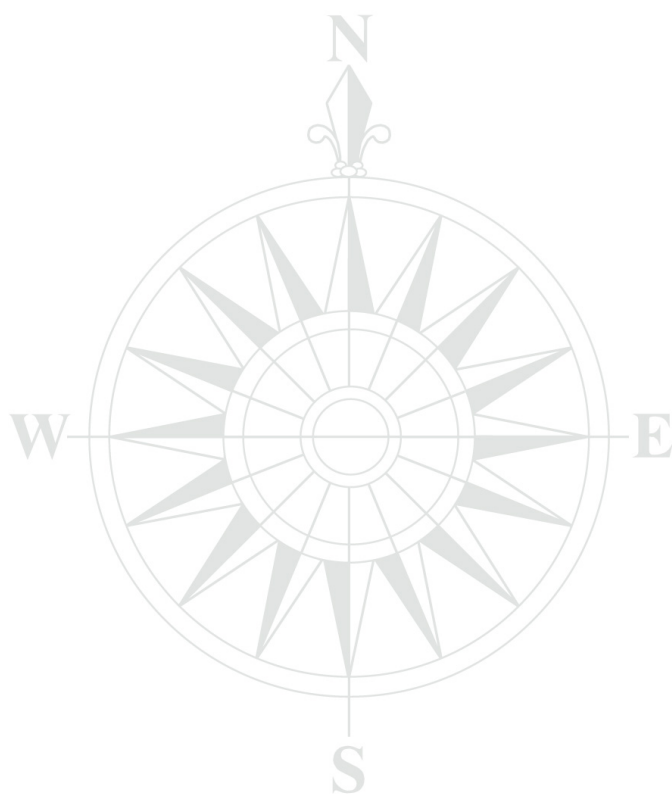


PolaRxS Reference Guide

Applicable to version 2.9.6 of the GNSS Firmware



PolaRxS Reference Guide

December 14, 2016

Applicable to version 2.9.6 of the GNSS Firmware

© Copyright 2000-2016 Septentrio nv/sa. All rights reserved.

Septentrio Satellite Navigation
Greenhill Campus, Interleuvenlaan 15i
3001 Leuven, Belgium

<http://www.septentrio.com>
support@septentrio.com
Phone: +32 16 300 800
Fax: +32 16 221 640

List of Contents

CONTENTS	6
LIST OF ACRONYMS.....	7
1 Firmware User Manual	11
1.1 QUICK START	12
1.1.1 Quick Start Equipment	12
1.1.2 Quick Start Procedure	12
1.2 How To.....	14
1.2.1 Connect to the Receiver	15
1.2.1.1 Via COM Ports	15
1.2.1.2 Via USB	15
1.2.1.3 Via a TCP/IP Port.....	15
1.2.1.4 Via a Web Browser	16
1.2.1.5 Via FTP	17
1.2.1.6 Connection Descriptors	17
1.2.2 Understand the Output of the Receiver.....	18
1.2.2.1 Proprietary Binary Output (SBF)	18
1.2.2.2 NMEA.....	18
1.2.2.3 RTCM and CMR	19
1.2.3 Output and Log SBF	20
1.2.4 Save the Configuration in Non-Volatile Memory	21
1.2.5 Configure the Receiver in DGPS/RTK-Base Mode	22
1.2.5.1 Static Base Station Mode	22
1.2.5.2 RTK Moving Base Station Mode.....	24
1.2.6 Configure the Receiver as a NTRIP Server	26
1.2.7 Configure an IP Server Port	27
1.2.8 Configure the Receiver in DGPS-Rover Mode	28
1.2.9 Configure the Receiver in RTK-Rover Mode	29
1.2.10 Use the TERRASTAR™ Corrections	30
1.2.11 Determine the Attitude of a Vehicle	31
1.2.11.1 Moving-Base Attitude.....	31
1.2.12 Configure the SBAS Operation.....	33
1.2.13 Log SBF or NMEA on the SD Memory Card	35
1.2.14 Log RINEX Files on the SD Memory Card	37
1.2.15 FTP Push SBF and RINEX files.....	38
1.2.16 Communicate with a Meteo Sensor.....	39
1.2.17 Generate a "Pulse Per Second" Signal	40
1.2.18 Time Tag External Events	41
1.2.19 Monitor the RF Spectrum	42
1.2.20 Manage Users	43

1.2.21 Upgrade the Receiver	44
1.2.22 Check the Capabilities of your Receiver	45
1.2.23 Check or Change the Permission File	46
1.2.24 Manage the Processor Load	47
1.3 OPERATION DETAILS	48
1.3.1 Channel Allocation and Signal Selection	48
1.3.2 Generation of Measurements	48
1.3.2.1 Pilot vs. Data Component	49
1.3.3 Time Management	49
1.3.3.1 Free-Running Clock	50
1.3.3.2 Clock Steering	50
1.3.4 Computation of Position, Velocity, and Time (PVT Solution)	51
1.3.4.1 SBAS Positioning	52
1.3.4.2 DGPS Positioning (Single and Multi-Base)	53
1.3.4.3 RTK Positioning	54
1.3.4.3.1 Pseudorange versus carrier phase: ambiguity	54
1.3.4.3.2 Carrier Phase Positioning	55
1.3.4.3.3 Integer Ambiguities (RTK-fixed)	55
1.3.4.3.4 Floating Ambiguities (RTK-float)	55
1.3.4.3.5 Moving Base	56
1.3.4.3.6 Antenna Effects	56
1.3.4.3.7 Practical Considerations	58
1.3.4.4 Transition between PVT Modes	58
1.3.4.5 Datum Transformation	58
1.3.4.5.1 Transformation to DGNSS/RTK Datum	58
1.3.4.5.2 Transformation to Local Datum	59
1.3.4.5.3 Projection to Plane Grid Representation	59
1.3.4.6 Precise Point Positioning	59
1.3.4.6.1 PPP Seeding	59
1.3.4.6.2 PPP Datum Offset	60
1.3.4.6.3 Tide Corrections	60
1.3.5 Receiver Autonomous Integrity Monitoring (RAIM)	60
1.3.5.1 Integrity Algorithm	62
1.3.5.2 Internal and External Reliability Levels	63
2 Command Line Interface	64
2.1 INTRODUCTION	65
2.1.1 Scope	65
2.1.2 Typographical Conventions	65
2.2 OUTLINE	66
2.2.1 Command Line Syntax	66
2.2.2 Command Replies	67
2.2.3 Command Syntax Tables	68
2.3 COMMAND LIST	70
2.3.1 Receiver Administration Commands	70
2.3.2 Authentication Commands	84
2.3.3 Tracking Configuration Commands	89
2.3.4 Navigation Configuration Commands	102
2.3.5 Receiver Operation Commands	138
2.3.6 Session Settings Commands	145

2.3.7	Input/Output Commands.....	148
2.3.8	RTCM v2.x Commands	173
2.3.9	RTCM v3.x Commands	180
2.3.10	CMR v2.0 Commands	191
2.3.11	Logging Commands.....	196
2.3.12	L-Band Commands	207
2.3.13	SBF List.....	215
3	SBF Block Interface	216
3.1	INTRODUCTION	217
3.1.1	Scope	217
3.1.2	Typographical Conventions	217
3.1.3	Change Log.....	217
3.2	SBF OUTLINE	219
3.2.1	SBF Block Header Format.....	219
3.2.2	List of SBF Block Names and Numbers	219
3.2.3	SBF Block Time Stamp (TOW and WNc)	222
3.2.4	Sub-blocks	223
3.2.5	Padding Bytes	223
3.2.6	SBF Revision Number	223
3.2.7	Do-Not-Use Value	224
3.2.8	Output Rate.....	224
3.2.9	Space Vehicle ID and GLONASS Frequency Number	225
3.2.10	Signal Type	225
3.2.11	Channel numbering	226
3.2.12	Decoding of SBF Blocks	226
3.3	SBF BLOCK DEFINITIONS	228
3.3.1	Measurement Blocks.....	228
3.3.2	Navigation Page Blocks.....	242
3.3.3	GPS Decoded Message Blocks	266
3.3.4	GLONASS Decoded Message Blocks	274
3.3.5	Galileo Decoded Message Blocks	280
3.3.6	Compass/BeiDou Decoded Message Blocks	293
3.3.7	QZSS Decoded Message Blocks	295
3.3.8	SBAS Decoded Message Blocks	297
3.3.9	Position, Velocity and Time Blocks.....	324
3.3.10	GNSS Attitude Blocks.....	383
3.3.11	Receiver Time Blocks.....	390
3.3.12	External Event Blocks.....	394
3.3.13	Differential Correction Blocks.....	406
3.3.14	L-Band Demodulator Blocks.....	415
3.3.15	Status Blocks	423
3.3.16	Miscellaneous Blocks	451
3.3.17	Deprecated or Obsolete Bocks.....	461
4	Index of Commands	462
A	Attitude Angles	473
A.1	VEHICLE REFERENCE FRAME	474
A.2	EULER ANGLES	475

B	NMEA Overview	476
B.1	PROPRIETARY NMEA SENTENCES	478
C	Error Messages	482

List of Acronyms

Abbreviation	Description
AGC	Automatic Gain Control
ARP	Antenna Reference Point
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation One
BeiDou	BeiDou navigation system
BGD	Broadcast Group Delay
CA	Coarse Acquisition
CMR	Compact Measurement Record
COG	Course Over Ground
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DGPS	Differential GPS
DHCP	Dynamic Host Configuration Protocol
DLL	Dynamically Linked Library
DNS	Domain Name Server
DOP	Dilution Of Precision
DVS	Data Validity Status
ECEF	Earth-Centered Earth-Fixed
EGNOS	European Geostationary Navigation Overlay System
ENU	east-north-up
FTP	File Transfer Protocol
GEO	Geostationary Earth Orbiter
GLONASS	Global Orbiting Navigation Satellite System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HDOP	Horizontal DOP
HERL	Horizontal External Reliability Level
HMI	hazardously misleading information
HPCA	HMI Probability Computation Algorithm

HPL	horizontal protection level
HS	Health Status
ICD	Interface Control Document
IEEE	Institute of Electrical and Electronics Engineers
IERS	International Earth Rotation Service
IF	Intermediate Frequency
IGP	Ionospheric Grid Point
IGS	International GPS Service
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IODC	Issue of Data - Clock
IODE	Issue Of Data Ephemeris
IP	Internet Protocol
IRNSS	Indian Regional Navigational Satellite System
ITRF	International Terrestrial Reference Frame
ITRS	International Terrestrial Reference System
LBand	L-Band Receiver
L1	L1 carrier
L2	L2 carrier
L2C	L2C code
LED	light emitting diode
LSB	Least Significant Bit
MDB	Minimum Detectable Bias
MIB	Management Information Base
MSB	Most Significant Bits
MT	Message Type
NATO	North Atlantic Treaty Organisation
NAV	Navigation
NAVSTAR	Navigation Satellite Timing And Ranging
NMEA	National Marine Electronics Association
P	P(Y) code
P1	P1 code
P2	P2 code

PDOP	Position DOP
PLL	Phase Locked Loop
PPP	Precise Point Positioning
PPS	Pulse Per Second
PRC	Pseudorange Correction
PRN	Pseudo Random Noise
PVT	Position, Velocity and Time
QZSS	Quasi-Zenith Satellite System
RAIM	Receiver Autonomous Integrity Monitoring
RINEX	Receiver Independent Exchange Format
RTCA	Radio Technical Commission for Aeronautics
RTCM	Radio Technical Commission for Maritime Services
RTK	Real-Time Kinematic
SBAS	Space-Based Augmentation System
SBF	Septentrio Binary Format
SF	Single Frequency
SIS	Signal In Space
SISA	Signal in Space Accuracy
SNMP	Simple Network Management Protocol
SV	Space Vehicle
SVID	Space Vehicle ID
TDOP	Time DOP
TOW	Time Of Week
UDRE	User Differential Range Error
UERE	User Equivalent Range Error
URA	User Range Accuracy
USB	Universal Serial Bus
UTC	Coordinated Universal Time
VDOP	Vertical DOP
VERL	Vertical External Reliability Level
VPL	vertical protection level
WAAS	Wide Area Augmentation System
WGS84	World Geodetic System 1984

WN	Week Number
WNc	Week number
XERL	External Reliability Levels
XOR	Exclusive OR
XPL	Horizontal or Vertical Protection Level

Chapter 1

Firmware User Manual

1.1 Quick Start

This chapter will help you to get quickly acquainted with your receiver by getting the first position fix.

1.1.1 Quick Start Equipment

You will need the following equipment to complete this quick start tutorial:

- An active GPS antenna. The standard antenna voltage compatible with the receiver is 5V.
- An antenna cable.
- The USB cable provided with your receiver.
- The power adaptor.
- A host computer which will be needed to operate your receiver and retrieve the data. In these quick-start instructions, you will learn how to use the RxControl program to monitor and control your receiver through the USB cable.
- The CD accompanying the receiver.

1.1.2 Quick Start Procedure

Step 1 Place the GNSS antenna horizontally in a place where the sky is not obstructed by buildings or trees. Connect the antenna via the antenna cable to the antenna port of the receiver.

Step 2 Install the RxTools software suite, which is to be found on the accompanying CD-ROM, and which includes various utilities to control the receiver and process the GNSS data. It is recommended to install all components of the installer (USB Driver, RxControl, Data Link, SBF Converter and RxLogger).

Step 3 Follow the instructions on the screen to install the USB driver. After a few seconds, the Windows USB driver will automatically create two virtual serial COM ports on your PC. If your operating system is Linux, only one virtual serial port is created by the default Linux driver.

Step 4 Start RxControl:

1. Open RxControl from the Start menu or by opening the shortcut on your desktop.
2. In the Connection Setup dialog from the Serial Connection drop down menu select Create New... and click the Next button.
3. Select one of the two virtual USB COM ports assigned to the receiver. They identify themselves as "Septentrio Virtual USB COM Port".
4. Enter any connection name and click the Finish button.
5. Wait a few seconds for the connection to take place.

Steps 2 to 4 have to be done only once: the next time you will restart RxControl, it will connect automatically by using previously entered connection parameters. Please always allow a few seconds between connecting the receiver and starting RxControl, in order for the USB driver to properly start up. To reconfigure your connection select Change Connection from the File menu and repeat steps 2-5 or click New Connection if you see a Connection Error dialog.

Step 5 After a few seconds, you will see the RxControl main window.

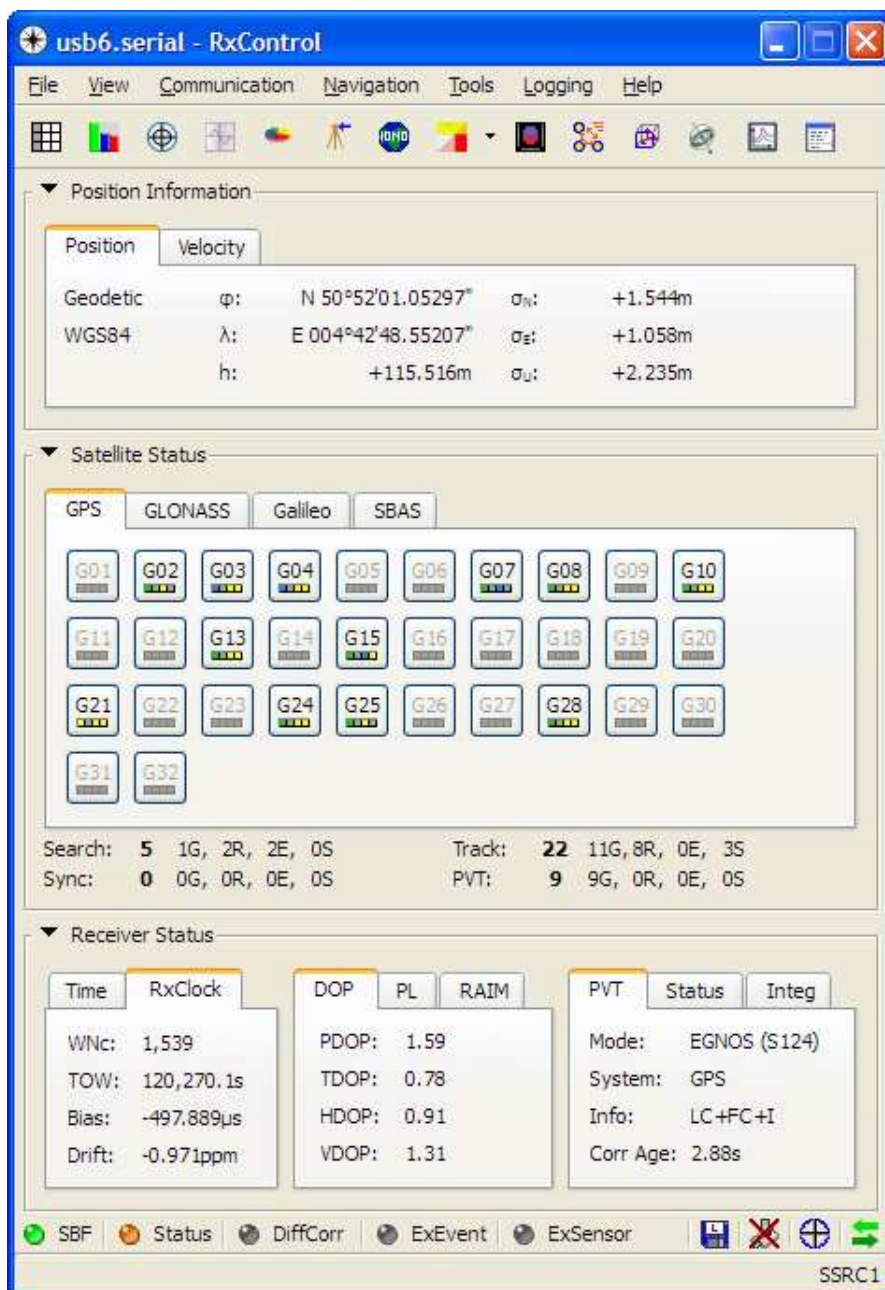


Figure 1.1-1: RxControl main window.

The central part of the RxControl main window shows the tracking status of the satellites in the different constellations supported by the receiver. Hover mouse over satellite buttons to see "Tool Tips" with more details. The position computed by the receiver is shown in the upper panel of the main window. The accuracy estimate for each position component is shown in the middle column.

Please consult the RxControl on-line help, under the *Help* menu, for more information.

1.2 How To...

This chapter contains step-by-step instructions to help you with typical tasks. It does not provide a complete overview of the receiver's operations, but rather an introduction to different operation modes. Please refer to the Command Line Interface Reference Guide for a complete description of the command set.

You can enter user commands in many different ways:

- Using the Data Link program provided in the RxTools suite, you can enter commands manually through one of the receiver input ports (see section 1.2.1). In this chapter, user commands are referred to by their full name for readability. When typing the command, you can always use the short mnemonic equivalent to save typing effort. For instance, instead of typing **setCOMSettings**, you can type **scs**. See the Command Line Interface Reference Guide to know the mnemonic equivalent of a given command.
- You can type commands or mnemonics in the console window of RxControl (menu *Tools > Expert Console*).
- You can also type commands or mnemonics from the web interface (*Admin > Receiver Management > Expert Console*).
- All commands can also be accessed graphically through menus in RxControl and in the web interface.



Depending on the capabilities of your particular receiver (see section 1.2.22), some of the features described here may not be supported.

1.2.1 Connect to the Receiver

1.2.1.1 Via COM Ports

A simple way to communicate with the receiver is to connect one of its COM-ports to a COM-port of your host computer. You can use the provided COM cable for this purpose. The default COM-port settings are:

Parameter	Value
baud rate	115200
data bits	8
parity	no
stop bits	1
flow control	none

The baud rate can be modified at any time by using the **setCOMSettings** command.

RxControl: [Communication > COM Port Settings](#)

Web Interface: [Configuration - Communication > COM Port Settings](#)

RxControl and Data Link can communicate with the receiver over a COM-port connection: select *Serial Connection* option when opening the connection to the receiver.

1.2.1.2 Via USB

The Windows USB driver provided with your receiver emulates two virtual serial ports, which can be used as standard COM ports to access the receiver. The Windows USB driver can be installed through the RxTools software suite. On Linux, the standard Linux CDC-ACM driver can be used to emulate one serial port. Most terminal emulation programs will make no distinction between virtual and native COM ports. Note that the port settings (baud rate, etc) for virtual serial ports are not relevant, and can be left in their default configuration in the terminal emulation program.

The main advantage of the USB connections with respect to the native COM ports is that they support a much larger bandwidth.

1.2.1.3 Via a TCP/IP Port

TCP/IP connections allow remote control of the receiver and are potentially much faster than serial connections. Up to eight independent TCP/IP connections can be opened in parallel through port 28784 (the port number can be changed with the command **setIPPortSettings**).

Over the Ethernet interface, the receiver can be configured for dynamic or fixed IP address allocation. The default is dynamic address allocation, using the DHCP protocol. The host-name is "ssrc-sxxxxxxxx", where xxxxxxxx consists of the last seven digits of the serial number of the receiver. The hostname is also printed on a label on the bottom side of the receiver.

Dynamic IP address allocation requires the availability of a DHCP server in your local network. In the absence of a DHCP server, or when a fixed IP address is desirable, it is possible to disable the DHCP client and use a fixed address. Switching between fixed and dynamic IP address allocation is typically done as follows, taking the fictitious example of setting the static IP address to 192.168.2.2, the netmask to 255.255.255.0 and the gateway to 192.128.2.1.

1. Specify the new IP settings with the command **setIPSettings**:
setIPSettings,Static,192.168.2.2,255.255.255.0,192.128.2.1 <CR>
[RxControl: Communication > Network Settings](#)
[Web Interface: Configuration - Communication > Network Settings](#)
2. Reset the receiver for the new settings to take effect:
exeResetReceiver,soft,none <CR>
[RxControl: File > Reset Receiver](#)
[Web Interface: Admin - Receiver Management > Reset Receiver](#)

RxControl and Data Link can communicate with remote receivers over a TCP/IP connection: select *TCP/IP Connection* option when opening the connection to the receiver.

1.2.1.4 Via a Web Browser

The receiver can be controlled and configured using a web browser. The hostname or fixed IP address is defined as explained in section 1.2.1.3. For example, if your receiver's hostname is `ssrc-sn1234567`, simply use the following URL in your preferred web browser:

http://ssrc-sn1234567

Figure 1.2-1: Web interface main window.

All *set*- and *exe*-user commands described in the Command Line Interface Reference Guide can be accessed from the web interface. You can also go to *Admin > Receiver Management > Expert Console* to manually enter commands and view replies.

By default, the web interface provides unrestricted read and write access to the receiver. This can be changed, as further explained in section 1.2.20 of this document.

1.2.1.5 Via FTP

FTP access allows to download (and delete if you have sufficient access rights) log files stored on the internal SD memory card. The hostname or fixed IP address is defined as explained in section 1.2.1.3. For example, if your receiver's hostname is `ssrc-sn1234567`, you could type the following URL in your preferred web browser to open a FTP session as anonymous user:

`ftp://ssrc-sn1234567`

The log files are found under the directory `ssn/SSRC3`.

By default, anonymous FTP users can download and delete files. This can be changed as explained in section 1.2.20.

See also section 1.2.13 for more details on internal logging.

1.2.1.6 Connection Descriptors

To direct output data to a given connection, the user has to specify the corresponding connection descriptor. Available connection descriptors are:

- COMx:** one of the native serial ports;
- USBx:** one of the virtual serial ports, built on top of the USB interface;
- DSKx:** one of the internal disks (or SD memory card);
- IP1x:** one of the TCP/IP connections;
- NTRx:** one of the NTRIP connections;
- IPSx:** one of the IP server connections (output only).
- IPRx:** one of the IP receive connections (input only).

For instance, to output the ASCII textual status screen to COM1, use:

`setDataInOut,COM1,,ASCIIIDisplay <CR>`

1.2.2 Understand the Output of the Receiver

The receiver outputs proprietary and standardized messages. Each proprietary message begins with a two-character identifier, which identifies the message type.

Proprietary messages	First two characters
ASCII command replies and command error notification	\$R
ASCII transmissions (e.g. periodic output of the status screen), terminated by a prompt. Two sub-types are defined: <ul style="list-style-type: none"> \$TD : ASCII display generated by the receiver; \$TE : event notification (e.g. receiver is shutting down). 	\$T
Formatted information blocks (e.g. formal command description)	\$-
SNMP' binary command replies (Septentrio proprietary)	\$&
Proprietary binary data (SBF)	\$@

Standardized messages
NMEA sentences
RTCM v2.x
RTCM v3.x
CMR v2.0

1.2.2.1 Proprietary Binary Output (SBF)

The binary messages conform to the SBF definition. The data are arranged in SBF blocks identified by block IDs. All the blocks begin with the SBF identifier \$@. Please refer to the SBF Reference Guide for a complete definition of SBF.

The benefit of SBF is compactness. This format should be your first choice if you wish to receive detailed information from the receiver.

The list of supported SBF messages on your particular receiver and firmware version can be found in the Command Line Interface Reference Guide.

SBF Converter, provided in the RxTools package is an intuitive GUI which allows SBF conversion into e.g. RINEX, KML, GPX or ASCII.

1.2.2.2 NMEA

The receiver can generate a set of approved NMEA sentences, which conform to the NMEA Standard⁽¹⁾. The benefit of the NMEA format is that it is standardized. Many electronic

⁽¹⁾ NMEA 0183, Standard for Interfacing Marine Electronic Devices, Version 2.30, National Marine Electronics Association, 1998

devices and software packages support NMEA. The drawback of NMEA is a relatively low level of detail. Appendix B provides a short overview of selected NMEA sentences.

NMEA output can be invoked with the **setNMEAOutput** command.

RxControl: Communication > Output Settings > NMEA Output > NMEA Output Intervals

Web Interface: Configuration - Communication > Output Settings > NMEA Output > NMEA Output Intervals

1.2.2.3 RTCM and CMR

If this feature is enabled in your receiver, the receiver can operate as DGPS and/or RTK base station and output the corresponding RTCM or CMR messages. The instructions to set the receiver in base station mode can be found in section 1.2.5. Appendix B provides a short overview of supported RTCM and CMR messages.

Note that the receiver supports the CMR+ and CMR-W format as input, but not as output.

It is possible to simultaneously output RTCM messages on one port, and CMR data on another port.

1.2.3 Output and Log SBF

The easiest way to log SBF blocks on your PC is to use the RxControl or RxLogger graphical programs, which are part of the RxTools suite. Under RxControl, go to the *Logging > Rx-Control Logging* menu to access the logging configuration window. Logging on the receiver's internal SD Memory Card is described in section 1.2.13 of this document.

In the following example, we show how to output SBF blocks using the command line interface. The example shows how to configure the receiver to output the `MeasEpoch` and `PVTCartesian` SBF blocks at 10 Hz and the `GPSNav` SBF block at its natural "OnChange" rate, i.e. when new GPS navigation data is available from a satellite. In this example, we will assume that these three blocks must be output to the USB2 connection.

1. First make sure that the USB2 connection is configured for SBF output (this is the default). In case this is not so, you should invoke:

```
setDataInOut,USB2,,+SBF <CR>
```

[RxControl: Communication > Input/Output Selection](#)

[Web Interface: Configuration - Communication > Input/Output Selection](#)

2. Scheduling SBF blocks for output is done by defining so-called "SBF streams". Up to 10 SBF streams can be defined by the user. A stream consists of a set of SBF blocks that need to be output at a given rate on a given connection descriptor. By default, all streams are empty, and no SBF blocks are output. For our example, we will need to use two streams: the first one for the `MeasEpoch` and `PVTCartesian` SBF blocks at a 10-Hz rate, and the second one for the `GPSNav` at the "OnChange" rate. Defining these SBF streams involves the `setSBFOutput` command:

```
setSBFOutput,Stream1,USB2,MeasEpoch+PVTCartesian,msec100 <CR>
```

```
setSBFOutput,Stream2,USB2,GPSNav,OnChange <CR>
```

[RxControl: Communication > Output Settings > SBF Output](#)

[Web Interface: Configuration - Communication > Output Settings > SBF Output](#)

If you want to output the same SBF blocks at the same rate on another connection, say, COM1, you will need to use two additional streams, for instance `Stream3` and `Stream4`:

```
setSBFOutput,Stream3,COM1,MeasEpoch+PVTCartesian,msec100 <CR>
```

```
setSBFOutput,Stream4,COM1,GPSNav,OnChange <CR>
```

3. To stop outputting SBF on a given connection, you can either redefine or empty the corresponding streams:

```
setSBFOutput,Stream1,USB2,none <CR>
```

```
setSBFOutput,Stream2,USB2,none <CR>
```

A second possibility is to disable all SBF messages on that connection:

```
setDataInOut,USB2,,-SBF <CR>
```

[RxControl: Communication > Input/Output Selection](#)

[Web Interface: Configuration - Communication > Input/Output Selection](#)

1.2.4 Save the Configuration in Non-Volatile Memory

The receiver configuration includes all the user-selectable parameters, such as the elevation mask, the PVT mode, the COM port settings,...

By default, the receiver starts up in its factory default configuration. The factory defaults for each of the receiver parameters are underlined for each argument of each command in the Command Line Interface Reference Guide.

At any time, it is possible to save the current receiver configuration into non-volatile memory, in order to force the receiver to always start up in that configuration. To do so, the following command should be entered:

exeCopyConfigFile,Current,Boot <CR>

RxControl: [File > Copy Configuration](#)

Web Interface: [Admin - Receiver Management > Copy Configuration](#)

To revert to the default setting where the receiver starts in the default configuration, you should use:

exeCopyConfigFile,RxDefault,Boot <CR>

1.2.5 Configure the Receiver in DGPS/RTK-Base Mode

The receiver can generate and output DGPS corrections or RTK data in the RTCM and CMR formats. The list of RTCM and CMR messages available on your particular receiver and firmware version can be found in the Command Line Interface Reference Guide (see the commands **setRTCMv2Output**, **setRTCMv3Output** and **setCMRv2Output**).

1.2.5.1 Static Base Station Mode

To configure the receiver in static base station mode, the following has to be done:

1. Connect the receiver to a survey-grade antenna at a fixed location.
2. For accurate and repetitive absolute positioning, you must provide the accurate coordinates of the antenna reference point (ARP). The ARP usually corresponds to the center of the bottom of the antenna (see also section 1.3.4.3.6). For example, assuming the WGS84 position of the ARP is 50.5°N, 4°E and its altitude above the WGS84 ellipsoid is 100m, use:

```
setStaticPosGeodetic,Geodetic1,50.5,4,100 <CR>  
setPVTMode,Static,,Geodetic1 <CR>
```

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration - Navigation > Positioning Mode > PVT Mode

If you are only interested in accurate determination of the base-rover baseline, with the absolute position of the rover being of lesser importance, accurate positioning of the base station is not required, and you may simply let the receiver determine its fixed position autonomously ("auto-base" mode), by typing:

```
setPVTMode,Static,,auto <CR>
```

3. For RTCM 3.x, the antenna information in message types 1007, 1008 and 1033 can be specified using the **setAntennaOffset** command, with the serial number as sixth argument, and the antenna type (called "antenna descriptor" in RTCM) as fifth argument (see also section 1.3.4.3.6). For instance:

```
setAntennaOffset,Main,,,"AT2775-54SW","5684" <CR>
```

RxControl: Navigation > Receiver Setup > Antennas

Web Interface: Configuration - Navigation > Receiver Setup > Antennas

4. Use the commands **setRTCMv2Interval**, **setRTCMv2IntervalObs**, **setRTCMv3Interval** or **setCMRv2Interval** to specify the message interval. The default interval is given in the description of these commands in the Command Line Interface Reference Guide. For instance, to change the default interval at which RTCM 2.x message type 3 is generated to 6 seconds, type:

```
setRTCMv2Interval,RTCM3,10 <CR>
```

RxControl: Communication > Output Settings > Differential Corrections > RTCMv2

Web Interface: Configuration - Communication > Output Settings > Differential Corrections > RTCMv2

5. Use the commands **setRTCMv2Formatting**, **setRTCMv3Formatting** or **setCMRv2Formatting** to specify the base station ID. If you are setting up multiple base stations, make sure to select a unique ID for each of them. For instance:

```
setRTCMv2Formatting,496 <CR>
```

RxControl: Communication > Output Settings > Differential Corrections > RTCMv2

Web Interface: Configuration - Communication > Output Settings > Differential Corrections > RTCMv2

6. Specify the baud rate of the serial port over which the RTCM or CMR messages have to be sent. For instance if the differential correction stream needs to be output on COM2 at 9600 baud, use:

setCOMSettings,COM2,baud9600 <CR>

RxControl: Communication > COM Port Settings

Web Interface: Configuration - Communication > COM Port Settings

7. It is recommended to enable code smoothing in order to mitigate propagation of multipath at the base station into the DGPS corrections and RTK data. For instance to smooth all pseudoranges with a smoothing length of 900s, use:

setSmoothingInterval,all,900 <CR>

RxControl: Navigation > Receiver Operation > Tracking and Measurements > Smoothing

Web Interface: Configuration - Navigation > Receiver Operation > Tracking and Measurements > Smoothing

8. According to the RTCM standard, an RTK base station must keep its clock error under 1.1 milliseconds. The CMR standard is even more stringent with a prescribed maximum clock error of 0.5ms (which is the receiver default). In case the receiver is not in its default configuration, you can restore the default setting by using:

setClockSyncThreshold,usec500 <CR>

RxControl: Navigation > Receiver Operation > Timing

Web Interface: Configuration - Navigation > Receiver Operation > Timing

9. By default, the receiver is configured to output all RTCM and CMR messages necessary for DGPS and RTK operation. In case the default has been modified, use the commands **setRTCMv2Output**, **setRTCMv3Output** or **setCMRv2Output** to specify which types of messages to enable for output. For instance, to output RTCM2.x messages 1 and 3 on COM2, use:

setRTCMv2Output,COM2,RTCM1+RTCM3 <CR>

RxControl: Communication > Output Settings > Differential Corrections > RTCMv2

Web Interface: Configuration - Communication > Output Settings > Differential Corrections > RTCMv2

10. The connection which needs to output the RTCM stream must be configured to do so. For instance, to enable RTCM 2.x output through COM2, use:

setDataInOut,COM2,,RTCMv2 <CR>

RxControl: Communication > Input/Output Selection

Web Interface: Configuration - Communication > Input/Output Selection

To stop transmitting RTCM messages, enter the following command:

setDataInOut,COM2,,none <CR>

RxControl: Communication > Input/Output Selection

Web Interface: Configuration - Communication > Input/Output Selection

Note that, even in static mode, the receiver computes a PVT solution to estimate the clock bias. Disabling the PVT, for example by using the **setSatelliteUsage** command, prevents the receiver from outputting RTK corrections.

1.2.5.2 RTK Moving Base Station Mode

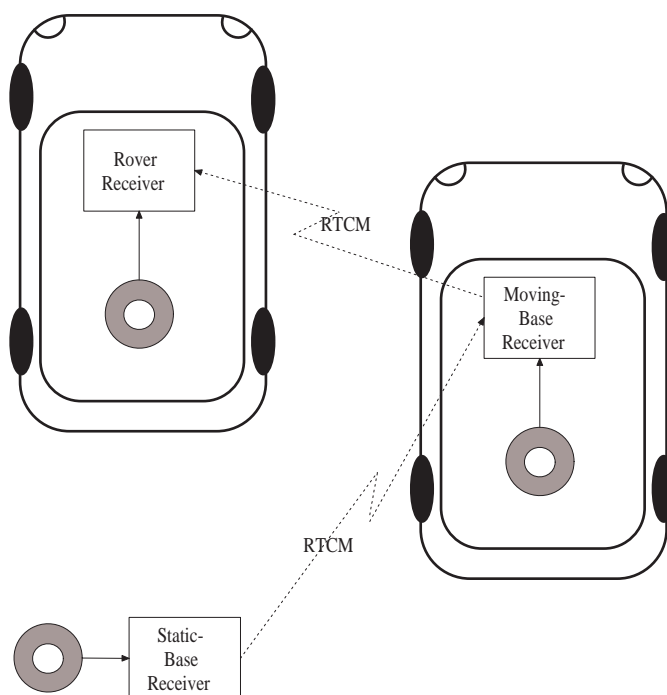


Figure 1.2-2: Example of a RTK moving-base configuration where the moving base receives RTCM corrections from a static base and transmits RTCM corrections to the rover.



Moving base station is only allowed in RTK mode (not in DGPS mode).

To configure the receiver in RTK moving base, follow the steps below:

1. The PVT engine must be set in one of the rover modes (standalone, DGPS, SBAS, RTK). The type of the PVT mode at the moving base station will determine the absolute position accuracy of the RTK rover receiver. On the other hand, the accuracy of the relative position of the rover with respect to the moving base is not influenced by the PVT mode at the moving base station.

For instance, to let the moving base station compute a simple standalone PVT, use the following:

setPVTMode, Rover, StandAlone <CR>

RxControl: [Navigation > Positioning Mode > PVT Mode](#)

Web Interface: [Configuration - Navigation > Positioning Mode > PVT Mode](#)

If accurate absolute and relative positioning of the rover is required, the moving base can operate in RTK-rover positioning mode and receive RTCM or CMR corrections from a static base station, as illustrated in figure 1.2-2. Refer to section 1.2.9 to configure the moving-base receiver in RTK-rover mode.

2. From now on, follow the same procedure as for a static base station, starting at step 3 of section 1.2.5.1 and taking into account the following recommendations:
 - RTCM v2.x and CMR are not supported for moving-base applications, use only RTCM v3.x in the link between the moving base and the rover.
 - To decrease the effect of extrapolation errors, use a short RTCM or CMR message interval (see the commands **setRTCMv3Interval** and **setCMRv2Interval**). In most cases, it is safe to set the interval to its minimum value of 0.1 seconds. If the RTCM or CMR messages are sent through a COM connection, make sure

that the baud rate is sufficient to support the high rate. A value of 115200baud is typical.

- In moving base, it is recommended to send the base position and observables at the same rate.

See also section 1.3.4.3.5 for more details on moving-base operation.

1.2.6 Configure the Receiver as a NTRIP Server

In the example below, we show how to configure the receiver to send RTCMv2 corrections to a NtripCaster using the following parameters:

- NtripCaster hostname: ntrip.example.com
- NtripCaster port: 2101
- User name/password for basic authentication: USER / PASSWD
- Mount Point: LEUV1

1. Configure one of the NTRIP connections (see section 1.2.1.6) for communication with the NtripCaster. Here, we assume that the first NTRIP connection (NTR1) is free and can be used for that purpose:

```
setNTRIPSettings,NTR1,Server,ntrip.example.com,2101,USER,PASSWD,LEUV1  
<CR>
```

RxControl: Communication > NTRIP Settings

Web Interface: Configuration - Communication > NTRIP Settings

2. By default, for RTCMv2, the receiver is configured to send message types 1, 3, 18, 19 and 22. This can be changed by using the command **setRTCMv2Output**. For instance, to send only message types 1 and 3 to the NtripCaster, use:

```
setRTCMv2Output,NTR1,RTCM1+RTCM3 <CR>
```

RxControl: Communication > Output Settings > Differential Corrections > RTCMv2

Web Interface: Configuration - Communication > Output Settings > Differential Corrections > RTCMv2

3. Enable the output of RTCMv2 data on the NTR1 connection:

```
setDataInOut,NTR1,,RTCMv2 <CR>
```

RxControl: Communication > Input/Output Selection

Web Interface: Configuration - Communication > Input/Output Selection

4. Closing the NTRIP connection is done with the following command:

```
setNTRIPSettings,NTR1,off <CR>
```

RxControl: Communication > NTRIP Settings

Web Interface: Configuration - Communication > NTRIP Settings

See also section 1.2.5 for more information on configuring the receiver as a base station.

1.2.7 Configure an IP Server Port

In this example, we show how to configure the receiver such that any client connecting to TCP/IP port 28785 will receive the NMEA GGA message at a 1-second interval.

1. Configure one of the IP server connections (see section 1.2.1.6) to listen to port 28785. Here, we assume that the first IP server connection (IPS1) is free:

setIPServerSettings, IPS1, 28785, TCP <CR>

RxControl: *Communication > Network Settings*

Web Interface: *Configuration - Communication > Network Settings*

2. Output the GGA NMEA message to the IPS1 connection, at a 1-Hz rate:

setNMEAOutput, Stream1, IPS1, GGA, sec1 <CR>

RxControl: *Communication > Output Settings > NMEA Output > NMEA Output Intervals*

Web Interface: *Configuration - Communication > Output Settings > NMEA Output > NMEA Output Intervals*

3. Make sure that NMEA output is enabled on the IPS1 connection. It is enabled by default, but in case your receiver is not in its default configuration, you should invoke:

setDataInOut, IPS1, , +NMEA <CR>

RxControl: *Communication > Input/Output Selection*

Web Interface: *Configuration - Communication > Input/Output Selection*

4. You will need to reset the receiver for the new IP server configuration to take effect:

exeResetReceiver, Soft, none <CR>

RxControl: *File > Reset Receiver*

Web Interface: *Admin - Receiver Management > Reset Receiver*

A way to check the IP server functionality is to enter the URL **http://ssrc-snnxxxxxx:28785** in your preferred web browser (replace *ssrc-snnxxxxxx* by the hostname of your particular receiver). You should see the NMEA GGA message coming every second.

Note that up to eight clients can concurrently connect to the same IP server port.

Conversely, the receiver can be configured to automatically receive data (for example differential corrections) from an IP server. This is done with the **setIPReceiveSettings**.

1.2.8 Configure the Receiver in DGPS-Rover Mode

The receiver can accept incoming DGPS corrections in the RTCM format from any of its connections. The list of DGPS correction messages supported by your particular receiver and firmware version can be found in the Command Line Interface Reference Guide (see the command **setRTCMv2Usage**). DGPS corrections can be received from a publicly available RTCM data provider, or from one or more Septentrio receivers configured as DGPS base.



In DGPS-rover mode, the base station must be static. Moving base stations are only supported in RTK-rover mode (see section 1.2.9).

To configure the receiver in DGPS-rover mode, the following has to be done:

1. The PVT processing needs to be configured to use DGPS corrections if they are available. This is the default, but in case your receiver is not in its default configuration, you should enter the following command:

setPVTMode, Rover, +DGPS <CR>

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration - Navigation > Positioning Mode > PVT Mode

2. When receiving differential corrections through one of the serial connections, make sure to configure the baud rate to match the baud rate of the incoming RTCM stream. For instance if the incoming RTCM stream is received through COM2 at a baud rate of 9600 baud, use:

setCOMSettings, COM2, baud9600 <CR>

RxControl: Communication > COM Port Settings

Web Interface: Configuration - Communication > COM Port Settings

For more details on DGPS refer to section 1.3.4.2.

1.2.9 Configure the Receiver in RTK-Rover Mode

The receiver can accept incoming RTK data in either the RTCM format or the CMR format from any of its connections. The list of RTK messages supported by your particular receiver and firmware version can be found in the Command Line Interface Reference Guide (see the commands **setRTCMv2Usage**, **setRTCMv3Usage** and **setCMRv2Usage**). RTK or CMR data can be received from a publicly available RTCM or CMR data provider, or from another Septentrio receiver configured as an RTK base station. The base station may be either static or moving. In static base mode, the receiver accepts RTCM 2.x, 3.x or CMR 2.0 messages. In moving-base mode, only RTCM 3.x and CMR 2.0 are supported.

To configure the receiver in RTK-rover mode, the following has to be done:

1. The PVT processing needs to be configured to use RTK data if they are available. This is the default, but in case your receiver is not in its default configuration, you should enter the following command:

setPVTMode,Rover,+RTK <CR>

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration - Navigation > Positioning Mode > PVT Mode

2. The type of base station (static or moving) has to be specified. For a static base, you should use:

setDiffCorrUsage,,,,,off <CR>

For a moving base, use:

setDiffCorrUsage,,,,,on <CR>

RxControl: Navigation > Positioning Mode > PPP and Differential Corrections

Web Interface: Configuration - Navigation > Positioning Mode > PPP and Differential Corrections

3. When receiving RTK data through one of the serial connections, make sure to configure the baud rate to match the baud rate of the incoming RTCM or CMR stream. For instance if the incoming stream is received through COM2 at a baud rate of 9600 baud, use:

setCOMSettings,COM2,baud9600 <CR>

RxControl: Communication > COM Port Settings

Web Interface: Configuration - Communication > COM Port Settings

When the baud rate is correctly configured, the receiver automatically detects the RTK data protocol (RTCM or CMR).

Please refer to section 1.3.4.3 for further details on the RTK positioning mode.

1.2.10 Use the TERRASTAR™ Corrections

Your receiver is fully compatible with the TERRASTAR-M and TERRASTAR-D services (see <http://terrastar.net>) offering high accuracy positioning without the need for a local base station.

In this section, we address the case where the primary positioning mode is RTK, and where the receiver uses the TERRASTAR-D precise point positioning (PPP) service to continue outputting high-accuracy positions during RTK outages.

Configuring your receiver for RTK/TERRASTAR-D positioning involves the following steps:

1. Activate the TERRASTAR-D service by following the procedure at <http://terrastar.net>.
2. Make sure that both RTK and PPP positioning modes are enabled in your receiver. This is the default, but in case the receiver is not in its default configuration, you can enable RTK and PPP by issuing the following command:
setPVTMode, Rover, +RTK+PPP <CR>
RxControl: Navigation > Positioning Mode > PVT Mode
Web Interface: Configuration - Navigation > Positioning Mode > PVT Mode
3. RTK positions are typically expressed in a regional datum which depends on your local RTK provider. Instead, TERRASTAR-D PPP positions relate to the global ITRF2008 reference frame. To avoid coordinate jumps each time the PVT engine switches between RTK and PPP modes, and to ensure accurate seeding of the PPP engine from RTK, the regional datum used by your RTK provider must be provided to the receiver. For example, in Europe, the RTK datum will often be ETRS89, and you would need to enter the following command:
setGeodeticDatum, ETRS89 <CR>
RxControl: Navigation > Positioning Mode > PPP and Differential Corrections
Web Interface: Configuration - Navigation > Positioning Mode > PPP and Differential Corrections
4. Enable the automatic seeding of the PPP engine with RTK positions:
setPPPAutoSeed, RTKFixed <CR>
RxControl: Navigation > Positioning Mode > PPP and Differential Corrections
Web Interface: Configuration - Navigation > Positioning Mode > PPP and Differential Corrections
5. Configure the RTK-rover mode as described in section 1.2.9.

1.2.11 Determine the Attitude of a Vehicle

Depending on your receiver model and permissions, there are different ways by which the receiver can compute the attitude (heading, pitch and roll angles) of a vehicle. See appendix A for a definition of the attitude angles.

1.2.11.1 Moving-Base Attitude

The heading and pitch of a vehicle can be derived from the orientation of the baseline between a base and a rover antenna when both antennas are attached to the vehicle. The base antenna is connected to a first receiver configured as RTK moving base station. The rover antenna is connected to a second receiver configured as RTK rover and accepting the RTCM stream from the first receiver. This is illustrated in figure 1.2-3.

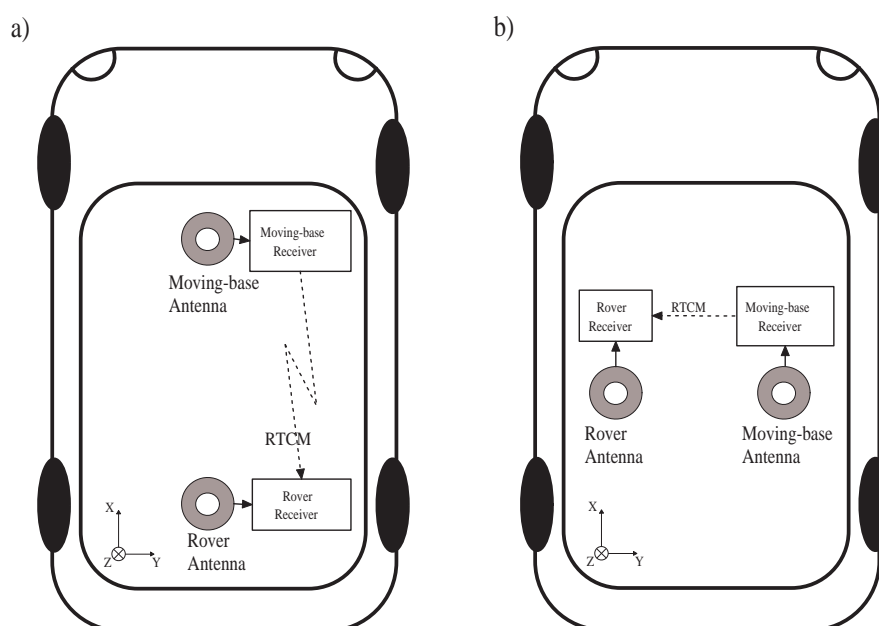


Figure 1.2-3: Moving-base attitude determination setup. a) default configuration.
b) example of non-default configuration.

To enable moving-base attitude determination, follow the following procedure:

1. Attach two antennas to your vehicle. The default antenna configuration is as depicted in figure 1.2-3 a). It consists in placing the antennas aligned with the longitudinal axis of the vehicle. If such configuration is not possible, you will have to provide additional information to the receiver, as explained below. For best accuracy, try to maximize the distance between the antennas, and avoid significant height difference between them.
2. Connect one of the antennas (preferably the one at the front of the vehicle) to the receiver that will serve as moving base. Connect the other to the receiver that will serve as rover. That latter receiver is the one where the heading and pitch will be computed.
3. Configure the moving-base receiver to send RTCM corrections to the rover. The procedure to do so is explained in section 1.2.5.2. Note that the RTCM stream may be transmitted either through a direct cable connection between the two receivers, or through a radio modem.

4. Configure the rover receiver to accept the RTCM corrections from the moving base, by following the steps in section 1.2.9.
5. By default, the attitude angles are computed assuming that the two antennas are aligned with the longitudinal axis of the vehicle, and that the moving-base antenna is in front of the rover antenna (see figure 1.2-3 a)). If you cannot place the antennas in such configuration, the reported attitude angles will be biased. There are two ways to remove these biases:
 - (a) The biases can be removed by telling the receiver where the moving-base antenna is located in the vehicle reference frame (see appendix A). This is done by specifying the coordinates of the baseline between the rover ARP and the moving-base ARP in the X, Y and Z directions. For example, in the configuration b) of figure 1.2-3, assuming that the distance between the antenna ARPs is 1 meter, you would issue (on the rover receiver):


```
setAntennaLocation, Base, manual, 0, 1, 0 <CR>
```

[RxControl: Navigation > Positioning Mode > GNSS Attitude](#)
[Web Interface: Configuration - Navigation > Positioning Mode > GNSS Attitude](#)
 - (b) Another way to remove the biases is to provide them to the receiver. For example, if the heading and pitch angles reported by the receiver have an offset of respectively 10 and 12 degrees with respect to their expected value, you can have the receiver compensate for that offset by using the following command:


```
setAttitudeOffset, 10, 12 <CR>
```

[RxControl: Navigation > Positioning Mode > GNSS Attitude](#)
[Web Interface: Configuration - Navigation > Positioning Mode > GNSS Attitude](#)
6. Specify that the attitude has to be computed in moving-base mode by issuing the following command in the rover receiver:


```
setGNSSAttitude, MovingBase <CR>
```

[RxControl: Navigation > Positioning Mode > GNSS Attitude](#)
[Web Interface: Configuration - Navigation > Positioning Mode > GNSS Attitude](#)

The attitude angles are available from the rover receiver in the `AttEuler` SBF block or in the HDT and HRP NMEA sentences.

1.2.12 Configure the SBAS Operation

Your receiver is by default configured to make optimal use of the wide-area corrections sent by these satellites. In case the receiver is not in its default configuration, you can reconfigure it as follows:

1. Make sure that SBAS positioning mode is enabled in the PVT. This is the default setting, but in case the receiver is not in its default configuration, you should use:

setPVTMode, Rover, +SBAS <CR>

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration - Navigation > Positioning Mode > PVT Mode

2. Make sure that the troposphere model is as prescribed by the RTCA DO 229 standard⁽²⁾. This is the default setting, but in case the receiver is not in its default configuration, you should use:

setTroposphereModel, MOPS, MOPS <CR>

RxControl: Navigation > Receiver Operation > Position > Atmosphere

Web Interface: Configuration - Navigation > Receiver Operation > Position > Atmosphere

3. It is recommended to leave the ionospheric model selection to `auto`. In particular, using the Klobuchar model in SBAS mode will lead to degraded performance and is not recommended.

setIonosphereModel, auto <CR>

RxControl: Navigation > Receiver Operation > Position > Atmosphere

Web Interface: Configuration - Navigation > Receiver Operation > Position > Atmosphere

4. By default, the receiver selects the SBAS satellite with the most SBAS corrections available. It is possible to force the receiver to select which SBAS satellite should provide the corrections to the PVT (and override the automatic selection by the receiver), and how to deal with subtleties of the SBAS navigation message. This is done by the **setSBASCorrections** command. For instance to only accept corrections from EG-NOS PRN126, use:

setSBASCorrections, S126 <CR>

RxControl: Navigation > Positioning Mode > SBAS Corrections

Web Interface: Configuration - Navigation > Positioning Mode > SBAS Corrections

5. Optionally, it is possible to include the range to SBAS satellites as an additional ranging source for the PVT. This is not done by default as the SBAS ephemeris accuracy is poor (100 m error). However to do so, use:

setSatelliteUsage, +SBAS <CR>

RxControl: Navigation > Advanced User Settings > PVT > Satellite Usage

Web Interface: Configuration - Navigation > Advanced User Settings > PVT > Satellite Usage

To compute a fully SBAS-aided position, the receiver has to receive and decode the following information:

- Long term corrections (corrections to the satellite orbit and clock as specified in the GPS ephemerides);
- Fast corrections (short term satellite clock error);
- Vertical ionospheric delays over the SBAS ionosphere grid surrounding the receiver position.

Due to the structure and order of the SBAS messages it can take up to 2.5 minutes before the long-term and fast corrections are available to the receiver and up to 5 minutes before the ionospheric grid is available. Hence it is normal that the receiver cannot yield an SBAS-aided position immediately after the lock on an SBAS satellite.

⁽²⁾ Minimum Operational Performance Standards for Global Positioning/Wide Area Augmentation System Airborne Equipment RTCA/DO-229C, November 28, 2001

For more details on SBAS positioning refer to section 1.3.4.1.

1.2.13 Log SBF or NMEA on the SD Memory Card

Enabling SBF or NMEA logging on the internal memory card involves the following steps:

1. By default, the receiver logs SBF blocks into a file named "log.sbf" and NMEA sentences into a file named "log.nma". You can specify any other fixed or auto-incrementing file name, or you can select the IGS/RINEX naming convention, where the file name automatically changes every fifteen minutes, hour, six hours or day. For instance, to let the receiver create daily files, use:

setFileNaming,DSK1,IGS24H <CR>

RxControl: Logging > Internal Logging Settings

Web Interface: Logging - Internal Logging Settings

If the file name you selected already exists, the receiver will append new data at the end of the existing file.

2. Use the command **setSBFOutput** to define which SBF blocks need to be logged (for NMEA, use **setNMEAOutput** instead), and at which interval (see also section 1.2.3). For instance, to log all SBF blocks necessary to build RINEX files, with the measurements and positions being output at a 10-s interval, use:

setSBFOutput,Stream1,DSK1,rinex,sec10 <CR>

RxControl: Communication > Output Settings > SBF Output

Web Interface: Configuration - Communication > Output Settings > SBF Output

The connection descriptor (see section 1.2.1.6) associated to the memory card is "DSK1".

3. Start the logging by enabling SBF and NMEA output to the DSK1 connection (it is enabled by default):

setDataInOut,DSK1, ,+SBF+NMEA <CR>

RxControl: Communication > Input/Output Selection

Web Interface: Configuration - Communication > Input/Output Selection

4. Once the logging session is finished, stop the logging by invoking:

setDataInOut,DSK1, ,-SBF-NMEA <CR>

RxControl: Communication > Input/Output Selection

Web Interface: Configuration - Communication > Input/Output Selection

On receivers with a "log button", you can press the log button to toggle SBF and NMEA logging on and off: each time the button is pressed, step 3 or 4 above is executed in turn.

There are different ways to download or delete files from the memory card:

- Using RxControl. Select *Logging > Download Internal Files* to download files to your computer, and *Logging > Remove Internal File* to remove a file from the memory card.

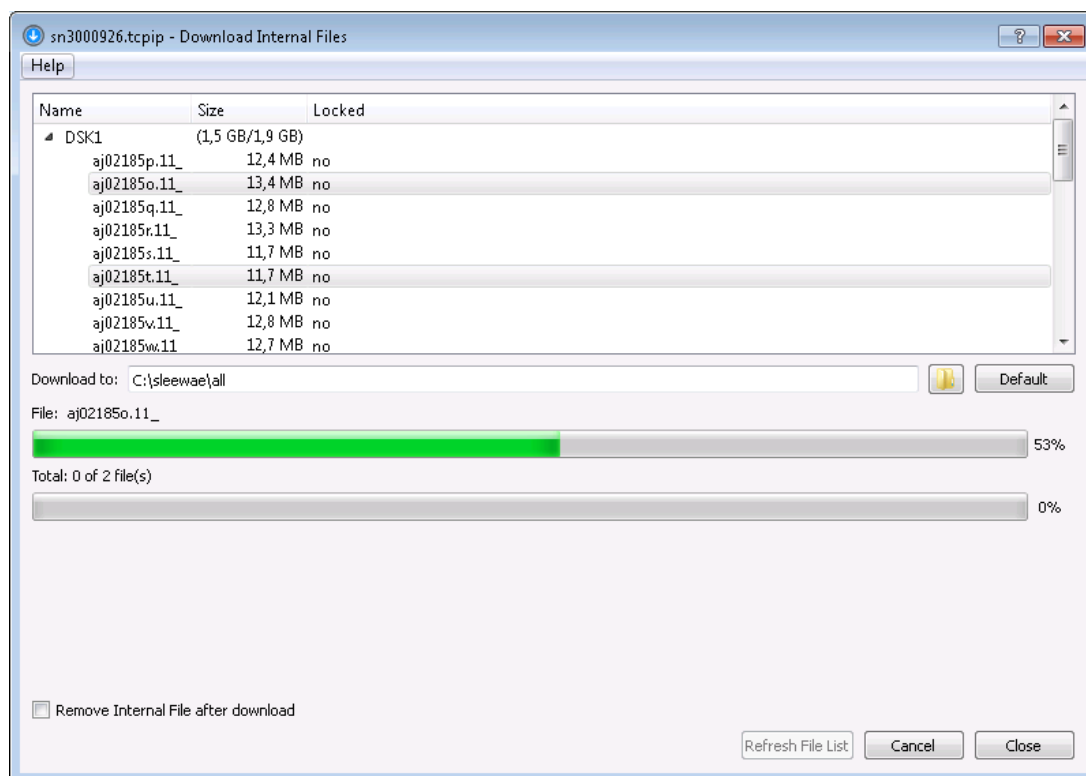


Figure 1.2-4: Download Internal Files from RxControl.

- Through FTP, see section 1.2.1.5.
- Using the web interface. The web interface allows you to view the contents of the memory card and to delete files (select the *Logging* tab, and then *Remove Internal File*). For file download, you will need to use FTP access (select the *Logging* tab, and then *FTP download*).
- By entering commands manually: the command **lstDiskInfo** prints the card contents and free space and the command **exeRemoveFile** can be used to remove a file.

1.2.14 Log RINEX Files on the SD Memory Card

The receiver can log the following RINEX file types on its internal SD memory card: O (observation), N (GPS nav), G (GLONASS nav), L (Galileo nav) and H (GEO nav). RINEX v2.10, v2.11 and 3.02 are supported.

Internal RINEX logging is typically configured as follows:

1. The RINEX file names follow the RINEX naming convention (*ssssdddf.yyt*), with the 4-character station name designator (*ssss*) being the first four characters of the marker name as specified with the **setMarkerParameters** command. For example, to set the station name designator to "LEUV", use:

setMarkerParameters, LEUV <CR>

RxControl: Navigation > Receiver Setup > Station Settings

Web Interface: Configuration - Navigation > Receiver Setup > Station Settings

2. Use the command **setRINEXLogging** to specify the file duration (fifteen minutes, one hour, six hours or one day), the observation interval and the type of observables to include in the RINEX file. For example, to generate daily RINEX files with the observation file containing only GPS L1CA data at a 30-s interval, use:

setRINEXLogging, DSK1, Hour24, sec30, GPSL1CA <CR>

RxControl: Logging > Internal RINEX Logging > RINEX Logging Options

Web Interface: Logging - Internal RINEX Logging > RINEX Logging Options

In this command, *DSK1* refers to the internal SD memory card.

3. The command **setDiskFullAction** specifies what to do when the SD memory card becomes full. For example, you could ask the receiver to automatically delete the oldest file to free up disk space. To do so, use:

setDiskFullAction, DeleteOldest <CR>

RxControl: Logging > Internal Logging Settings

Web Interface: Logging - Internal Logging Settings

Instead of logging RINEX files inside the receiver, you can also convert a SBF file to RINEX using the **sbf2rin** program or the **SBFConverter** graphical tool.

1.2.15 FTP Push SBF and RINEX files

It is possible to configure the receiver to automatically send internally-logged SBF or RINEX files to a remote FTP server (FTP Push). This is done with the **setFTPPushSBF** and **setFTPPushRINEX** commands respectively.

For example, to automatically FTP RINEX files to the directory `mydata/rin` on the remote server `myftp.com`, with username `myname` and password `mypwd`, you would enter the following command:

setFTPPushRINEX, myftp.com, mydata/rin, myname, mypwd <CR>

RxControl: [Logging > Internal RINEX Logging > RINEX FTP Push Options](#)

Web Interface: [Logging - Internal RINEX Logging > RINEX FTP Push Options](#)

To FTP push SBF files to the same location, you would use:

setFTPPushSBF, myftp.com, mydata/rin, myname, mypwd <CR>

RxControl: [Logging > Internal RINEX Logging](#)

Web Interface: [Logging - Internal RINEX Logging](#)

Note that all files are put in the same remote directory (`mydata/rin` in this example), even if they are internally logged in daily directories. FTP push does not create daily folders on the remote server.

1.2.16 Communicate with a Meteo Sensor

The receiver can send periodical queries to an external sensor (such as a meteo sensor) connected to one of its serial ports, and log the replies from that sensor. In the following example, we show how to retrieve meteo data every 10 seconds from a meteo sensor connected to the receiver's COM2 port.

1. Tell the receiver which command to use to query the external sensor, and the interval at which this command must be sent to the sensor. For instance, for a MET3/MET4-compatible sensor, the command `*0100P9<CR><LF>` queries the meteo data. Assuming you want to get meteo data at a 10-second interval, enter the following command:

```
setPeriodicEcho, com2, A:*0100P9%%CR%%LF, sec10 <CR>
```

[RxControl: Communication > Output Settings > Periodic Echo Message](#)

[Web Interface: Configuration - Communication > Output Settings > Periodic Echo Message](#)

2. Enable unformatted ASCII input on COM2 (to receive the replies from the meteo sensor):

```
setDataInOut, COM2, ASCIIIn <CR>
```

[RxControl: Communication > Input/Output Selection](#)

[Web Interface: Configuration - Communication > Input/Output Selection](#)

The replies from the meteo sensor (containing the temperature, pressure and humidity) are available in the `ASCIIIn` SBF block. Refer to section 1.2.3 to know how to output and log SBF blocks.

You can convert a SBF file containing `ASCIIIn` SBF blocks to RINEX using the `sbf2rin` program or the `SBFConverter` graphical tool. These tools support MET3/MET4 compatible sensors.

1.2.17 Generate a "Pulse Per Second" Signal

The receiver is able to generate an x-pulse-per-second (xPPS) signal aligned with either GPS, Galileo or GLONASS system time, or with UTC, or with the internal receiver time. The interval between pulses can be set to 0.1, 0.2, 0.5, 1, 2, 5 or 10 seconds.

By default, the PPS is a positive pulse of which the leading edge is synchronous with the second boundaries of the time system selected with the **setTimingSystem** command (GPS or Galileo). Check the Hardware Manual for the voltage and the duration of the pulse.

The command **setPPSPParameters** can be used to synchronize the PPS with UTC, GLONASS or the internal time, or to alter the PPS interval and polarity. For instance, to synchronize the PPS with UTC and have one pulse every ten seconds, use:

setPPSPParameters,sec10,, ,UTC <CR>

[RxControl: Navigation > Receiver Operation > Timing](#)

[Web Interface: Configuration - Navigation > Receiver Operation > Timing](#)

When the PPS output is configured in GPS, Galileo, GLONASS or UTC mode, the antenna and cable delays will cause the PPS to be offset from its correct position. The third argument of the **setPPSPParameters** command can be used to specify the overall antenna and cable delay, in order to allow the receiver to compensate for them.

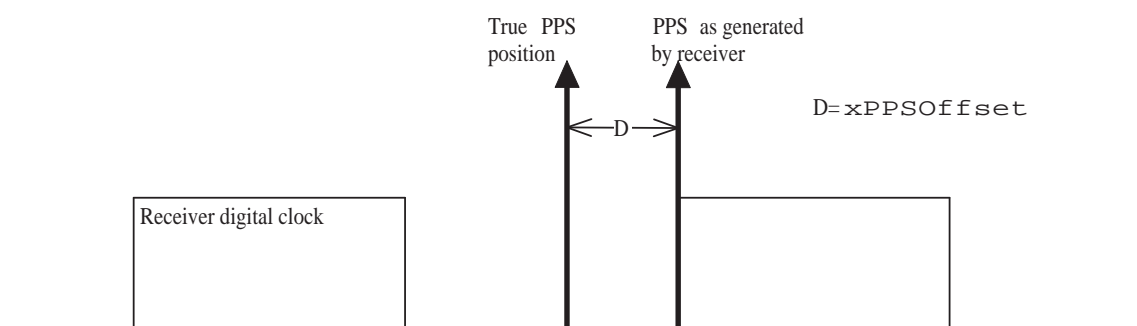


Figure 1.2-5: xPPS output granularity.

Although the position of the PPS pulse is computed accurately by the receiver, the actual pulse is generated at the nearest "tick" of the internal receiver digital clock, as illustrated in the figure above. This leaves an offset (noted "D" in the figure) between the true xPPS pulse and the one actually generated by the receiver. This offset can reach a few nanoseconds. It is available in real-time in the **xPPSOffset** SBF block.

To be able to align its xPPS output with the GNSS system time, the receiver needs a fresh estimate of the GNSS time from its PVT solution. If the last PVT solution is older than a prescribed timeout (set by the **setPPSPParameters** command), no PPS pulse is generated. In addition, to align its PPS with UTC, the receiver needs to have received the UTC offset parameters from the satellite navigation messages. If these parameters are not available and the user has requested to align the xPPS with UTC, no xPPS pulse is generated too.

1.2.18 Time Tag External Events

The receiver can time-tag electrical level transitions on its EventX inputs with an accuracy of 20ns.

By default, the receiver reacts on low-to-high transitions. You can use the **setEventParameters** command to react on falling edge instead:

setEventParameters,EventA,High2Low <CR>

RxControl: [Navigation > Receiver Operation > Timing](#)

Web Interface: [Configuration - Navigation > Receiver Operation > Timing](#)

Upon detection of a transition, the receiver can output the time and/or the position at the instant of the event (see the external event SBF blocks in the SBF Reference Guide).

The following constraints must be observed to ensure proper event detection:

- There must be no more than four events in any interval of 50 milliseconds, all event pins considered.
- The minimum time between two events on the same EventX input must be at least 5ms.

Missed events are flagged by the **MISSEDEVENT** bit in the **ReceiverStatus** SBF block.

1.2.19 Monitor the RF Spectrum

You can monitor the RF spectrum using the spectral analyzer in RxControl (go to the *View* > *Spectral View* menu). This allows to detect the presence of interferences in the GNSS bands.

In the example shown below, a narrowband interference at 1180 MHz is clearly visible.

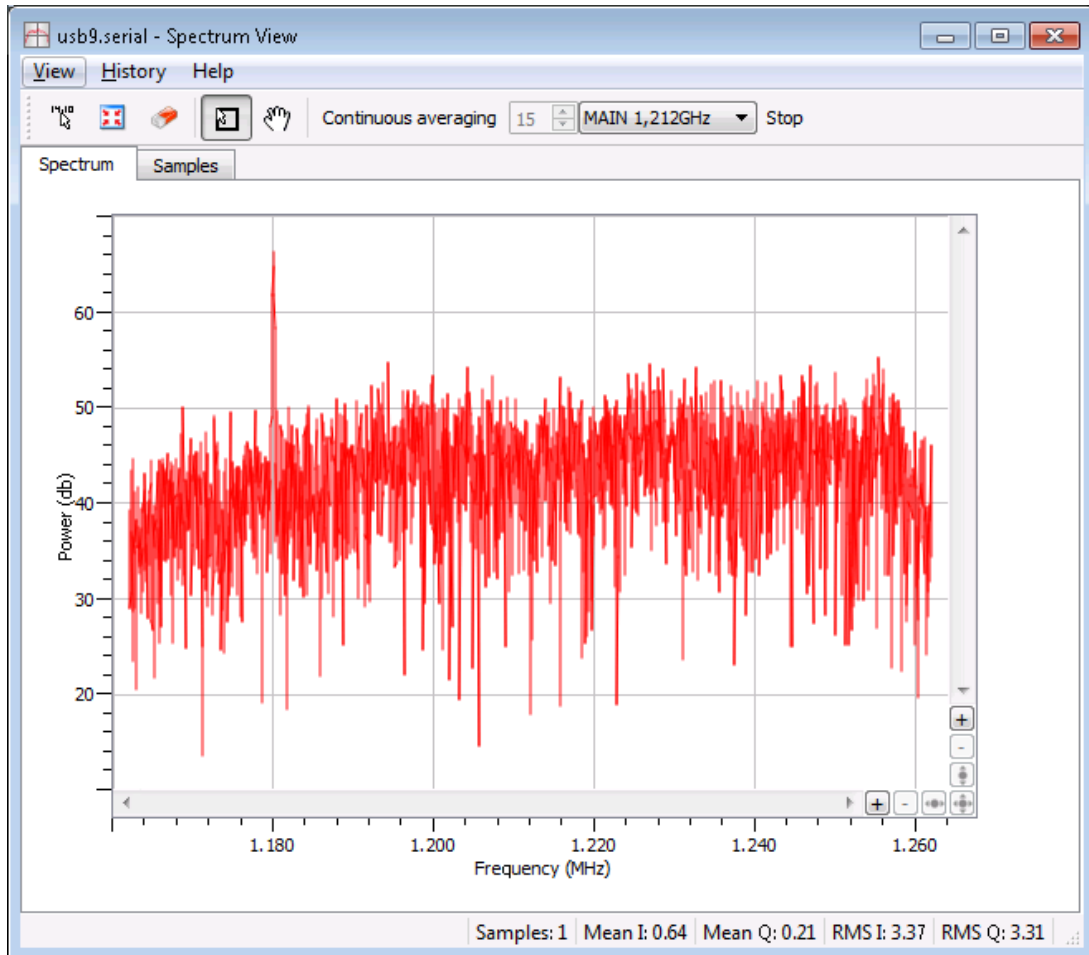


Figure 1.2-6: Spectral Analyser functionality of RxControl.

The spectrum is computed from baseband samples taken at the output of the receiver's analog to digital converters. These samples are available to the users in the `BBSamples` SBF block.

1.2.20 Manage Users

When connecting to the receiver, users can remain "anonymous", or can log in using the **login** command. What anonymous users can do depend on the connection type. By default, anonymous users have full control of the receiver. This default configuration can be changed with the command **setDefaultAccessLevel**. For example, to prevent anonymous access to the web interface and to the ftp server, you would use: **setDefaultAccessLevel, none, none <CR>**

RxControl: *File > User Management*

Web Interface: *Admin - Receiver Management > User Management*

To perform actions not allowed to anonymous users, you first need to authenticate yourself by entering a user name and a password through the **login** command. The list of user names and passwords and their respective access level is managed with the **setUserAccessLevel** command. Login fails if the provided user name or password is not in that list.

Logged-in users are granted one of the following access levels: "User" or "Viewer". The "User" level allows full control of the receiver, while the "Viewer" level only allows to view the configuration.

The following explains how to add or delete a user.

1. Check the current user list by entering the following command:

getUserAccessLevel <CR>

RxControl: *File > User Management*

Web Interface: *Admin - Receiver Management > User Management*

The reply to this command looks like:

```
UserAccessLevel, User1, "admin", "R46NCG", User
UserAccessLevel, User2, "", "", Viewer
UserAccessLevel, User3, "", "", Viewer
...
```

2. In the example shown above, only one user is defined, **User1** with user name **admin**. For security reasons, the password shown here **R46NCG** is random and does not correspond to the actual password. It can be seen that the level of access of the **admin** user is "User": that particular user has full control of the receiver. To add a new user "john" with password "abc123" and to give full access to that user, select a free user index, e.g. **User2** in the above example, and type:
setUserAccessLevel, User2, john, abc123, User <CR>
3. You can add up to eight users in this way. Deleting a user involves entering an empty string ("") as user name and password. For example, to delete the "admin" user from the above list, use:

setUserAccessLevel, User1, "", "" <CR>

The user list also applies to FTP accesses. FTP users having the "User" access right are allowed to delete files from the SD memory card via FTP, while "Viewer" FTP users can only download files.

1.2.21 Upgrade the Receiver

Upgrading the receiver is the process of installing a new GNSS firmware, a new permission file (see section 1.2.23) or a new antenna calibration file (see section 1.3.4.3.6).

Upgrading the GNSS firmware can clear the receiver configuration stored in non-volatile memory (see section 1.2.4). Please make sure to reconfigure your receiver (e.g. baud rate settings, elevation masks, LBAS1 access code, etc) after an upgrade.



Do not switch power off during the upgrade procedure.

Septentrio upgrade files have the extension “.suf”. There are several ways to upgrade the receiver:

1. By double clicking the “.suf” file. This should launch the RxUpgrade program.
2. By using the RxControl graphical interface (go to the *File* menu).
3. From the web interface (go to *Admin > Receiver Management*). This requires to log in as a user with the "User" access level (see section 1.2.20).
4. By commanding the receiver to upgrade itself by fetching the upgrade file from a remote FTP server. This is done with the command **exeFTPUpgrade**.
[RxControl: File > Upgrade Receiver from FTP](#)
[Web Interface: Admin - Receiver Management > Upgrade Receiver using FTP](#)
5. By manually downloading upgrade files to the receiver. This upgrade procedure is explained below.

To manually upgrade the receiver, follow this procedure:

1. Reset the receiver into upgrade mode by entering the following command:
exeResetReceiver, Upgrade, none <CR>
[RxControl: File > Reset Receiver](#)
[Web Interface: Admin - Receiver Management > Reset Receiver](#)
2. Wait till the receiver outputs the string: “Ready for SUF download ...”. From that moment on, the receiver is waiting for an upgrade file to be downloaded. The file download must start within 200 seconds, otherwise the receiver will restart in normal mode.
3. Download the upgrade file to the receiver. Any of the receiver connections can be used (COM, USB or IP). Make sure to send the file in binary mode, i.e. without changing its contents. During the download, the receiver outputs a progress indicator at regular interval.
4. At the end of the download, the receiver automatically executes the upgrade instructions and restarts with the new firmware version. You can check the firmware version by entering the following command:
lif,Identification <CR>

Before executing the upgrade instructions, the receiver checks the integrity of the downloaded file. If the file is corrupted, or is not a valid upgrade file, the receiver discards it and restarts in normal mode.

If the download is interrupted for any reason, the receiver will restart in normal mode after a timeout period of 200 seconds.

1.2.22 Check the Capabilities of your Receiver

The capabilities of your receiver are defined by the set of enabled features. The capabilities depend on the hardware, the current firmware version and the current set of permissions. Permissions are further explained in section 1.2.23.

The command **getReceiverCapabilities** lists the capabilities. You can also check them using the web interface (go to *Admin - Receiver Interface > Permitted Capabilities*) or Rx-Control (go to *Help > Receiver Interface* and select the *Permitted Capabilities* tab):

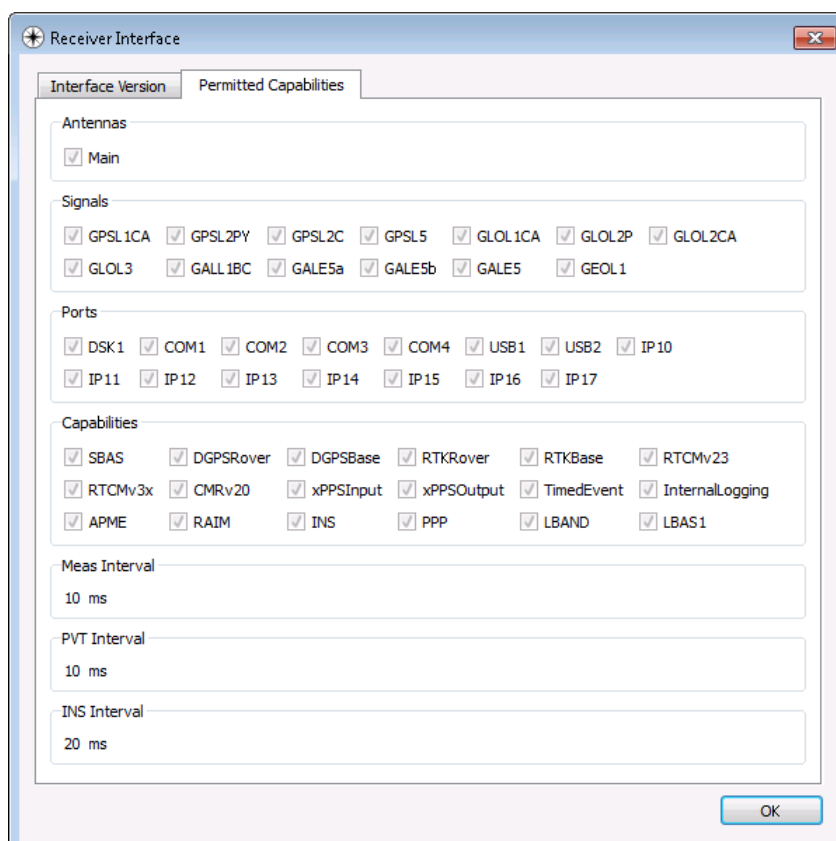


Figure 1.2-7: Example of receiver capabilities.

1.2.23 Check or Change the Permission File

The permission file lists which optional features (such as GLONASS, Galileo, RTK, ...) are permitted on your receiver, for how long they are permitted and in which region they are permitted.

The permission file is stored in the receiver's non-volatile memory, and can be checked with the command **1stInternalFile, Permissions**, or with RxControl by clicking *Help > Receiver Permissions*, or with the web interface (select the *Admin* tab, and then *Receiver Interface*).

Note that, for a given feature to be enabled in the receiver, it must be permitted and the hardware and firmware version must support it. See also section 1.2.22.

Each receiver is delivered with a permission file applicable to that receiver only. To enable new options, the user can order a new permission file to Septentrio, and install it on his/her receiver using the standard upgrade procedure (see section 1.2.21).

1.2.24 Manage the Processor Load

The processor load (also referred to as the CPU usage) is reported in the `ReceiverStatus` SBF block and can be viewed on the main window of RxControl and of the web interface. Receiver operation becomes unreliable when the CPU usage gets higher than 90%. CPU overload may lead to software errors, and it is typical that the `SOFTWARE` error bit in the `ReceiverStatus` SBF block be set if that happens (use the command `!stInternalFile, Error` to reset that bit).

High processor load is typically observed during high-rate RTK or multi-base DGPS operation.

A number of actions can be undertaken to free up CPU resources:

- Lower the output rate of SBF blocks (see the `setSBFOutput` command), and only enable those blocks needed for your application.
- Limit the number of satellites being tracked, for instance by increasing the elevation mask (`setElevationMask` command).
- Disable SBAS or GLONASS tracking if SBAS or GLONASS is not required for your application, using the `setSatelliteTracking` command.
- Disable the tracking of signals not needed for your application (e.g. GPS L2C), using the `setSignalTracking` command.
- Disable the "ASCIIIDisplay" output with the `setDataInOut` command: this display is primarily meant for temporary inspection of the receiver operation and for debugging.

1.3 Operation Details

This Chapter describes the key processes implemented in the receiver and explains how they can be configured.

1.3.1 Channel Allocation and Signal Selection

The receiver automatically allocates satellites to tracking channels up to the limit of the number of channels. It is possible to override this automatic channel allocation by forcing a satellite to a given channel by using the **setChannelAllocation** command. Also, a subset of satellites or a whole constellation can be disabled with the **setSatelliteTracking** command.

For each satellite, the receiver tries to track all signal types enabled with the **setSignalTracking** command. For example, if that command enables the GPSL1CA, GPSL2PY and GLOL1CA signals, GPS satellites will be tracked in dual-frequency mode (GPSL1CA and GPSL2PY) and GLONASS satellites will be tracked in single-frequency mode (GLOL1CA only). It is a good practice to only enable those signal types that are needed for your application to avoid wasting tracking channels.

1.3.2 Generation of Measurements

For each tracked GNSS signal, the receiver generates a "measurement set", mainly consisting of the following observables:

- a pseudorange in meters;
- a carrier phase in cycles;
- a Doppler in Hertz;
- a carrier-to-noise ratio in dB-Hz.

All data in a measurement set, and all measurement sets are taken at the same time, which is referred to as the "measurement epoch". All the measurement sets taken at a given measurement epoch are output in a **MeasEpoch** SBF block.

Several commands affect the way the receiver produces and outputs measurements:

- The **setHealthMask** command can be used to filter out measurements from unhealthy satellites: these measurements will not be used by the PVT algorithm, nor will they be included in the **MeasEpoch** SBF block.
- To further reduce the code measurement noise, the receiver can be ordered to smooth the pseudorange by the carrier phase. This technique, sometimes referred to as a "Hatch filtering", allows to reduce the pseudorange noise and multipath. It is controlled by the **setSmoothingInterval** command and is disabled by default.
- The **setMultipathMitigation** command can be used to enable or disable the mitigation of multipath errors in the pseudorange. It is enabled by default.

For advanced applications or in-depth signal analysis, the **MeasExtra** SBF block contains various additional data complementing the **MeasEpoch** SBF block. Among other things, this block reports the multipath correction applied to the pseudorange (allowing one to recompute the original pseudorange), and the observable variances.

1.3.2.1 Pilot vs. Data Component

Most modern GNSS signals consist of two components: a so-called pilot component and a data component. For such signals, the measurements are based on the pilot component for optimal performance. In particular, the reported C/N_o value is that of the pilot component only.

For all signals having a pilot and a data component, the table below indicates which component is tracked by Septentrio receivers. Note that your particular receiver model may not support all of these signals.

Signal	Signal component being used for measurement generation
GPS/QZSS L2C	L2C-L
GPS/QZSS L5	L5-Q
Galileo L1	L1-C
Galileo E6	E6-C
Galileo E5a	E5a-Q
Galileo E5b	E5b-Q
Galileo E5AltBOC	E5AltBOC-Q
GLONASS L3	L3-Q

1.3.3 Time Management

All time tags in the receiver refer to the receiver time scale. The receiver is designed in such a way that the receiver time is kept as close as possible to the selected GNSS system time (GPS or Galileo as prescribed by the **setTimingSystem** command). Internally, the receiver time is kept in two counters: the time-of-week counter in integer milliseconds (TOW) and the week number counter (WNc). WNc counts the number of complete weeks elapsed since January 6, 1980 (even if the selected GNSS system time is Galileo). The TOW and WNc counters are reported in all SBF blocks.

The synchronization of TOW and WNc with the GNSS system time involves the following steps:

- Upon powering up the receiver, TOW and WNc are assumed unknown, and set to a "Do-Not-Use value" in the SBF blocks.
- The transmission time-of-week and week number are coded in the GPS or Galileo navigation messages:
 - As soon as the first time-of-week is decoded from the GPS or Galileo signal-in-space (SIS), the TOW counter is initialized to within 20 ms of GNSS system time and starts counting. This is also the time when the receiver starts generating measurements.
 - As soon as the week number is decoded from the GPS or Galileo SIS (which can be either simultaneously with the time-of-week, or several seconds later), the WNc counter is set and starts counting.
- After the first position and time fix has been computed (for which measurements from at least 4 satellites are required), TOW is set to within X milliseconds of GNSS time.

This is done by introducing a jump of an integer number of milliseconds in the TOW counter. X is the maximal allowed offset between the receiver time and GNSS time, and is set by the **setClockSyncThreshold** command (by default, $X=0.5\text{ms}$). This initial clock synchronization leads to a simultaneous jump in all the pseudorange and carrier phase measurements.

The level to which the receiver time is synchronized with the GNSS system time is given by three status bits (**TOWSET**, **WNSET** and **FINETIME**) available both in the **ReceiverTime** SBF block and the **ReceiverStatus** SBF block.

The receiver clock can be configured in free-running mode, or in steered mode using the command **setClockSyncThreshold**.

1.3.3.1 Free-Running Clock

In free-running mode, the receiver time slowly drifts with respect to GNSS time. The receiver continuously monitors this time offset: this is the clock bias term computed in the PVT solution, as provided in the **RxCkBias** field of the **PVTCartesian** and **PVTGeodetic** SBF blocks. A clock jump of an integer number of milliseconds is imposed on the receiver clock each time the clock bias exceeds X milliseconds by an absolute value (X is set by **setClockSyncThreshold**). This typically results in a saw-tooth profile similar to that shown in Figure 1.3-1. In this example, $X=0.5\text{ms}$ and each time the clock bias becomes greater than 0.5ms , a jump of 1ms is applied.

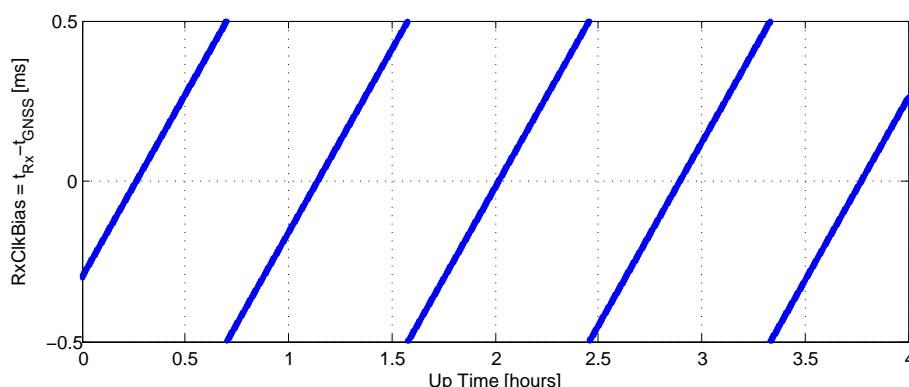


Figure 1.3-1: Example of the evolution of the receiver time offset with respect to the GNSS time in free-running mode.

When a receiver clock jump occurs, all measurements jump simultaneously. For example, a clock jump of 1ms will cause all the pseudoranges to jump by $0.001\text{s} \times \text{velocity_of_light} = 299792.458\text{m}$. The jump is applied on both the pseudoranges and the carrier phase measurements, and hence will not be seen on a code-minus-phase plot.

The cumulated clock jumps since the last reset of the receiver is reported in the **CumClkJumps** field of the **MeasEpoch** SBF block.

1.3.3.2 Clock Steering

In steered mode, the receiver time is continuously steered to GNSS time to within a couple of nanoseconds. In the example of Figure 1.3-1, if the user would have enabled clock steering

one hour after start up of the receiver, the clock bias would have been like in Figure 1.3-2 below.

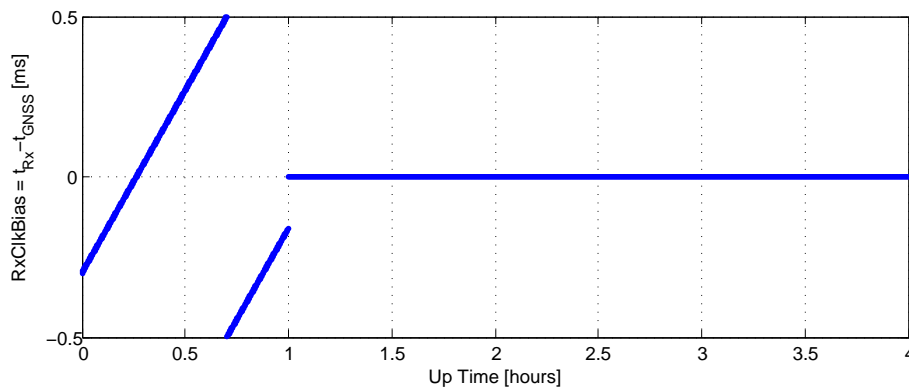


Figure 1.3-2: Effect of clock steering on the clock bias (clock steering enabled at an up time of 1 hour).

Clock steering accuracy is dependent on the satellite visibility, and it is recommended to only enable it under open-sky conditions.

Bit 3 of the `CommonFlags` field of the `MeasEpoch` SBF block indicates whether clock steering is active or not.



Note for the users of a GNSS constellation simulator

When using a constellation simulator, make sure to set the simulation time after January 01, 2010. The receiver time will be incorrect before that date.

For correct time determination, it is mandatory to reset the receiver before every (re)start of the simulation.

1.3.4 Computation of Position, Velocity, and Time (PVT Solution)

The receiver computes the position and velocity of its antenna, and the time offset of the receiver based on the pseudoranges, the Doppler measurements and, if applicable, the differential corrections.

The availability of the PVT depends on:

- the number of available pseudoranges and Doppler measurements, equal to the number of tracked satellites, or a subset of them as specified by the `setSatelliteUsage` command;
- the number of valid sets of broadcast ephemerides, which are needed to compute the position, velocity, and clock bias for each tracked satellite;
- the number of valid sets of fast and long-term SBAS corrections and their age in the case of SBAS-aided positioning;
- the number of valid differential corrections and their age in the case of DGPS/RTK positioning.

A position fix requires a minimum of 4 tracked satellites with associated ephemerides. When only 3 satellites are available or in case of bad satellite geometry (large DOP), the receiver will compute a 2D position fix assuming that the ellipsoidal height is the same as for the latest 3D fix. The mode of position fix is reported by the `Mode` field in the PVT-related SBF blocks. If less than 3 satellites are available, the receiver does not compute a position.

When a PVT solution is not available, PVT-related SBF blocks are still output with all the numeric fields set to Do-Not-Use values, and with the `Error` field set to indicate the source of the problem.

The accuracy of the PVT depends on:

- The signal level: measurements with a C/N_0 of 32 dB-Hz will exhibit considerably more noise than measurements with a C/N_0 of 52 dB-Hz. Hence it is recommended to use a high quality antenna.
- The geometry of the satellite constellation expressed in the DOP values: these values indicate the ratio of positional errors to range errors and are computed on the basis of the error propagation theory. When the DOP is high, the accuracy of positioning will be low.
- The number of available satellites: the more satellites are available, the lower the DOP. Measurement redundancy also enables better outlier detection.
- Multipath errors on the pseudorange measurements: multipath errors can be largely attenuated by enabling the APME multipath mitigation method (see `setMultipathMitigation`) and/or using code smoothing (see `setSmoothingInterval`).
- The PVT mode as set by the `setPVTMode` command: the user can select between the following modes, listed in the order of increasing accuracy: standalone, SBAS, DGPS and RTK.
- The data available to compute ionospheric delays (see `setIonosphereModel`).
- The choice of the dynamics model: if the dynamics parameter set by the `setReceiverDynamics` command does not correspond to the actual dynamics of the receiver platform, the position estimation will be sub-optimal.

The a-posteriori accuracy estimate of the computed position is reported in the variance-covariance matrix, which comes in the `PosCovCartesian` and `PosCovGeodetic` SBF blocks. This accuracy estimate is based on the assumed measurement noise model and may differ from actual errors due to many external factors, most of all multipath.

By default, the pseudoranges from the geostationary SBAS satellites are not used in the PVT solution due to the lower quality of the SBAS ephemerides and pseudoranges. However, for applications where satellite availability is expected to be low, it could be beneficial to allow their use in the PVT computation. This can be done by using the `setSatelliteUsage` command.

1.3.4.1 SBAS Positioning

SBAS, which stands for 'Space Based Augmentation System', enables differential operation over a large area with associated integrity information. System errors are computed from a dataset recorded over a continental area and disseminated via a geostationary satellite. The operation of SBAS is documented in the RTCA DO 229 standard. SBAS improves over DGPS corrections, in that it provides system corrections (ionosphere corrections and

ephemeris long-term corrections) next to range corrections (the "fast corrections" in the DO 229 terminology).

The receiver provides an SBAS-aided position when it has sufficient satellites with at least fast and long-term corrections. The corrections are used as long as their applicability has not timed out. During the time-out interval the receiver applies correction degradation using the information received in message type (MT) 07 and 10.

The receiver will attempt to optimize the selection of the SBAS correction provider based on the number of corrections available. For example when it has only 4 corrections from EGNOS but 8 corrections from WAAS the receiver will use the WAAS satellite even though it may be located in the EGNOS service area.

The PVT propagates the correction variances into a horizontal protection level (HPL) and a vertical protection level (VPL). These protection levels indicate the expected user error with an integrity of 10^{-7} . Note that these protection levels only refer to the signal-in-space errors. Local effects such as severe multipath are not considered into the HPL/VPL computation.

If the service provider transmits MT27 and MT28, the receiver can detect when it is located outside the service area and adjust the PVT accuracy accordingly. Without these messages the receiver has no means of knowing the extent of the service area.

The DO 229 standard defines two operation modes for SBAS positioning: en-route and precision approach. As the integrity requirements for precision approach are significantly higher, the HPL/VPL values in this mode are higher and, more importantly, the time-out interval of the corrections is shorter, which can lower the availability of a position. The default operation of the receiver is en-route, and the user has the choice to select precision approach using the **setSBASCorrections** command.

An SBAS provider can transmit MT00 to reset the data transmission in case of severe errors. However, this message is also transmitted for test purposes. For proper operation during a test phase (such as ESTB), it is recommended to ignore the MT00, which can be done using the **setSBASCorrections** command.

The `GEOCorrections` SBF block contains all the corrections and their variances as used in the PVT computation. This block allows for a detailed analysis of the SBAS PVT computation in the receiver.

1.3.4.2 DGPS Positioning (Single and Multi-Base)

DGPS (Differential GPS) reduces the effect of GNSS system errors by the use of range corrections. GNSS system errors such as orbit and atmospheric errors are highly correlated within an area of several kilometers. This can be exploited by computing the pseudorange errors with respect to one or more known locations and by transmitting these errors to nearby users. The receiver can be configured as a DGPS rover, in which it accepts range corrections, or as base in which it computes range corrections.

Local errors at base stations, such as multipath, will propagate into the rover position. Hence a high quality antenna should be used and care should be taken in the choice of the location of the base station(s). Furthermore any error in the base coordinates will translate in the rover position.

To work in DGPS rover mode, the receiver requires the reception of differential corrections. The format of these corrections is standardized in RTCM.

Note that the receiver takes the τ_{gd} parameter transmitted by the GPS satellites into account during the computation of the pseudorange corrections, as prescribed in v2.2 and v2.3 of the RTCM standard. The RTCM standard version 2.1 is ambiguous in this respect: it does neither prescribe nor discourage the use of τ_{gd} . The receiver can be configured in both modes using the command **setRTCMv2Compatibility**.

If the received RTCM stream contains corrections from multiple base stations, the receiver will compute a multi-base DGPS solution, unless the user has forced the usage of a particular base station with the command **setDiffCorrUsage**. Be aware that multi-base DGPS can quickly overload the receiver processor if the number of base stations is large.

1.3.4.3 RTK Positioning

RTK, which stands for "Real-Time Kinematic", is a carrier phase positioning method where the carrier phase ambiguities are estimated in a kinematic mode: it does not require static initialization.

To work in RTK mode, the receiver requires the reception of RTK messages. Both the RTCM and the CMR message formats are supported. The base station providing these RTK messages can be either static or moving. Multiple-base RTK is not supported: by default, the receiver selects the nearest base station if more than one base station is available.

In RTK mode, the absolute position is reported in the `PVTCartesian` or `PVTGeodetic` SBF blocks, and the baseline vector is reported in the `BaseVectorCart` and `BaseVectorGeod` SBF blocks.

1.3.4.3.1 Pseudorange versus carrier phase: ambiguity

Pseudoranges typically have a thermal noise in the decimeter range. The resulting position accuracy is in the meter range if multipath and orbit errors are taken into account. On the other hand, the phase measurements from the carrier signal are very precise, with a millimeter-level precision.

However, phase measurements are by nature ambiguous. Consider the dial of a clock as an analogue: if only the big hand would be available on the dial we would only know how many minutes have gone by. Only by counting the hour crossovers every 60 minutes we could gain the knowledge of the current hour. GPS carrier phase measurements behave in the same way: we only know the current phase but do not know the total number of wavelengths which make up the range to the satellite: the carrier phase contains an ambiguity. To actually use the carrier phase measurement as a satellite range, this ambiguity has to be resolved.

Summing up, pseudorange measurements are low accuracy absolute ranges to GPS satellites, while carrier phase measurements are high precision relative ranges to satellites. By estimating the ambiguity, the carrier phase measurements are turned into high-accuracy satellite ranges, and the low accuracy pseudoranges are not needed for positioning.

1.3.4.3.2 Carrier Phase Positioning

To use the high accuracy of the carrier phase measurements, error sources such as broadcast ephemeris errors, satellite clock errors and atmospheric delay must be eliminated as much as possible. This is achieved by performing differential positioning: by differencing the phase measurements with those of a receiver at a nearby location. The common errors are eliminated and the position can be accurately estimated with respect to this base station. This requires two receivers which are connected by a data link. One receiver (the base) is located at a known location and transmits its position and measurements to another receiver (the rover) which is placed at the location of interest. Standardized data format for this measurement exchange are RTCM 2.2 and higher or CMR. Thanks to this standardization, measurements from publicly available reference stations can also be used, eliminating the need for a second receiver. The distance between the roving receiver and the reference station will be the driving factor to make the choice between a dedicated and a public base station: as the baseline length increases, the common errors will start to decorrelate.

Due to the differential nature of phase positioning, the unknown ambiguities of phase measurements become integer. This is the key to the accuracy of carrier phase positioning: if the exact integer value of the ambiguity is known, phase measurements can be used as highly accurate satellite ranges. If the ambiguity cannot be estimated as an integer, the ambiguity will absorb errors that did not completely cancel in the differential application, such as multipath.

1.3.4.3.3 Integer Ambiguities (RTK-fixed)

Under normal circumstances the receiver will compute the integer ambiguities within several seconds and yield an RTK-fixed solution with centimeter-level accuracy. The less accurate pseudorange measurements will not be used. As long as no cycle slips or loss-of-lock events occurs, the carrier phase position is readily available.

RTK with fixed ambiguities is also commonly referred to as phase positioning using 'On-The-Fly' (OTF) ambiguity fixing. The RTK positioning engine of the receiver uses the LAMBDA method⁽³⁾ developed at Delft University, department of Geodesy.

1.3.4.3.4 Floating Ambiguities (RTK-float)

When data availability is low (no L2 data or low number of satellites) or when the data are not of sufficient quality (high multipath), the receiver will not fix the carrier phase ambiguities to their integer value, but will keep them floating. At the start of the RTK-float convergence process, the position accuracy is equal to that of code-based DGPS. Over the course of several minutes the positional accuracy will converge from several decimeters to several centimeters as the floating ambiguities become more accurate.

⁽³⁾ Teunissen, P.J.G., and C.C.J.M. Tiberius (1994) Integer least-squares estimation of the GPS phase ambiguities. Proceedings of International Symposium on Kinematic Systems in Geodesy, Geomatics and Navigation KIS'94, Banff, Canada, August 30-September 2, pp. 221-231.

1.3.4.3.5 Moving Base

In RTK, the base station does not necessarily need to be static. In some applications, one is interested in the relative positioning of two moving vehicles. In that case, both base and rover receivers are mounted on moving platforms and the RTK engine computes the baseline between them. If both base and rover receivers are mounted on the same vehicle, the baseline can be used to determine the orientation of the vehicle (see section 1.2.11.1). If accurate absolute positioning is required in addition to relative positioning, the moving base receiver can operate in RTK mode and get RTK correction from a fixed base station (see section 1.2.5.2).

With the command `setDiffCorrUsage`, the rover receiver must be informed that the base is moving. The baseline coordinates and orientation is contained in the `BaseVectorCart` and `BaseVectorGeod` SBF blocks.

Due to delays in the generation and transmission of the RTK data (base station position and measurements) from the base to the rover, the RTK data has a certain "age" when received by the rover. When operating with a moving base station, the RTK engine is of the "low-latency" type. This means that, when the rover computes its RTK position at time t_0 , it extrapolates the most recently received RTK data from the base to time t_0 . The accuracy of this extrapolation, and hence the accuracy of the final RTK solution, degrades with the age of the RTK data. Therefore it is essential that the base sends its position and measurements at a sufficient rate.

The default rate of 1 Hz is adequate in the case of a static base station, but is generally too low for a moving base with a non-constant velocity. For its extrapolation, the rover assumes a constant velocity of the base. If the base is subject to an acceleration a , the extrapolation error for an age Δt is given by $a\Delta t^2/2$. Even for moderate values of acceleration, it is apparent that the error will rapidly grow (e.g. it is 50 cm for an acceleration of 0.1g and an age of 1 second). In moving base operation, it is therefore recommended to set the RTK data rate to its maximum allowed value of 10 Hz.

Not only the RTK data rate, but also the communication link latency is important. Especially in moving base, it is essential to have a low-latency communication link between base and rover. To avoid old data to corrupt the RTK solution, the rover discards any RTK data of which the age exceeds a prescribed threshold (see the `setDiffCorrUsage` command). The default threshold value is 20 seconds. For moving base, it is recommended to reduce this value to 5 seconds.

1.3.4.3.6 Antenna Effects

To achieve the highest precision in RTK operations, it is essential to take antenna effects into account.

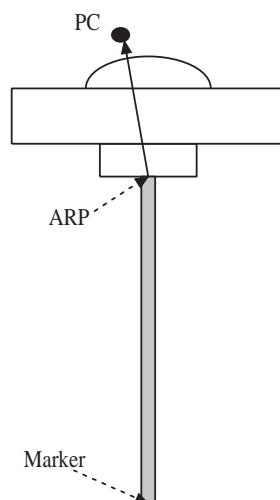


Figure 1.3-3: Antenna mount.

The GNSS measurements (pseudoranges and carrier phases observables) refer to a theoretical point in space called the phase center (noted PC in figure 1.3-3). The position of this point is dependent on the elevation of the satellite and on the frequency band. It varies with time and it is different for L1 and L2. The phase center variation can reach a few centimeters.

If no correction is applied, the computed position refers to an "average" phase center with no easy link with the antenna physical element. This average phase center fluctuates with time and cannot be used for accurate millimeter-level positioning.

For high-precision positioning, the GNSS measurements need to be corrected in such a way that they all refer to a common and stable point in space. That point is referred to as the antenna reference point (ARP). For convenience, it is usually selected at the center of the bottom surface of the antenna. The National Geodetic Survey has calibrated the offset from the PC to the ARP as a function of the elevation and of the frequency band for a large number of geodetic-grade antennas. NGS publishes calibration tables that can be downloaded from the following URL:

<http://www.ngs.noaa.gov/ANTCAL>.

The antenna naming convention in such table is the one adopted by the IGS Central Bureau.

The receiver has a similar table in its non-volatile memory. This table can be upgraded following the standard upgrade procedure as described in section 1.2.21 (the upgrade file is named `ant_info.suf`). To let the receiver compensate for the phase center variations and compute the ARP position, the user must specify the type of his/her antenna using the `setAntennaOffset` command. If the antenna is not specified, or the antenna type is not present in the antenna calibration file, the receiver cannot make the distinction between phase center and ARP, and the position accuracy is slightly degraded, especially in the height component.

The point to be positioned is the "marker" (see figure 1.3-3). The offset between the ARP and the marker is a function of the antenna monumentation. It must be measured by the user and specified with the `setAntennaOffset` command.

The absolute position reported in the `PVTCartesian` and `PVTGeodetic` SBF blocks is always the marker position.

The base-to-rover baseline coordinates in the `BaseVectorCart` and `BaseVectorGeod` SBF blocks is from ARP to ARP unless the receiver is not able to properly compensate for the phase center variation at base or rover. Details on this is to be found in the description of these blocks in the SBF Reference Guide.

1.3.4.3.7 Practical Considerations

The reasons for possible low accuracy or availability of the RTK position are:

- Multipath;
- Ionosphere decorrelation;
- Loss-of-lock;
- L2 availability;
- RTCM/CMR availability.

To ensure high accuracy and availability, care must be taken that the above error sources have as little impact as possible. This can be achieved by using survey-grade antennas and choosing a suitable location for the base station with an unobstructed view of the sky. Since low-elevation satellites are more prone to loss-of-lock and multipath, it is also recommended to use an elevation mask of 10 degrees. In moving-base applications, it is recommended to keep the baseline length short (<1km).

1.3.4.4 Transition between PVT Modes

Whenever possible, the transitions from a more accurate PVT mode to a less accurate PVT mode are smooth. For example, when switching from RTK to DGPS mode, the position does not exhibit a sudden jump, but slowly degrades from RTK to DGPS accuracy.

1.3.4.5 Datum Transformation

By default the datum to which the coordinates refer depends on the positioning mode. For standalone, PPP and SBAS positioning for example, the coordinates refer to a global datum: WGS84 or ITRS. When using DGNSS or RTK corrections from a local DGNSS/RTK provider, the datum is determined by the coordinates of the base station.

Recent realisations of WGS84 and ITRS are closely aligned and the difference can be neglected in most cases. The receiver considers them equivalent. However, regional datums may significantly differ from WGS84/ITRS, which may lead to coordinate jumps when switching between different positioning modes.

1.3.4.5.1 Transformation to DGNSS/RTK Datum

It is possible to avoid this datum shift by configuring the receiver to transform all coordinates to the regional datum used by the RTK base stations. This is done with the `setGeodeticDatum` command. The receiver knows the transformation parameters applicable to the most common datums (e.g. ETRS89 or NAD83), but user datums can also be defined with the `setUserDatum` command.

Coordinates in the `PVTCartesian` and `PVTGeodetic` SBF blocks refer to the datum selected in `setGeodeticDatum`. The datum can be checked by decoding the `Datum` field of these blocks.

1.3.4.5.2 Transformation to Local Datum

Some RTK networks provide transformation parameters in RTCM v3.x message types 1021 to 1023. These transformation parameters are meant to transform the RTK coordinates from the network regional datum into a local datum. The local coordinates are reported in a specific SBF block: `PosLocal`.

The following conditions must be met for the receiver to provide a valid position in the `PosLocal` SBF block:

- the usage of RTCM v3.x MT1021-1023 must be enabled by the command `setRTCMv3Usage` (these messages are enabled by default);
- the complete set of applicable datum transformation messages must have been received;
- the position must be in the area of validity of the transformation parameters.

1.3.4.5.3 Projection to Plane Grid Representation

If the RTK network provides coordinate projection parameters, the receiver reports the plane grid coordinates (northing and easting) in the `PosProjected` SBF block.

The following conditions must be met for the receiver to provide valid coordinates in the `PosProjected` SBF block:

- the usage of RTCM v3.x MT1021-1027 must be enabled by the command `setRTCMv3Usage` (these messages are enabled by default);
- projection parameters must be sent out by the service provider (using either MT1025, MT1026 or MT1027).
- the complete set of applicable datum transformation and projection messages must have been received;
- the position must be in the area of validity of the transformation/projection parameters.

1.3.4.6 Precise Point Positioning

Precise Point Positioning (PPP) provides high accuracy positioning without the need for a local base station. PPP uses precise satellite orbit and clock corrections computed by a global network of reference stations and broadcast in real time by geostationary satellites in the L band.

1.3.4.6.1 PPP Seeding

PPP provides centimeter-level position accuracy, but suffers from a relatively long convergence time that can reach 15 to 20 minutes depending on the local multipath environment. The convergence time can be dramatically reduced by feeding the known position into the

PPP engine. This process is referred to as PPP seeding, and the position fed into the PPP engine is called the PPP seed. The receiver supports two seeding modes:

Manual Seeding: in manual seeding, the user provides the accurate marker position (see section 1.3.4.3.6 for a definition of the marker position) to the PPP engine by using the `exePPPSetSeedGeod` command.

Automatic Seeding: the receiver can be configured to automatically seed the PPP engine from its current DPGS or RTK position. Automatic seeding is configured with the `setPPPAutoSeed` command.

1.3.4.6.2 PPP Datum Offset

By default, PPP positions are expressed in ITRS (which ITRF realization of ITRS depends on the PPP service provider), while RTK positions refer to the regional datum used by your RTK provider. To avoid coordinate jumps each time the PVT engine switches between RTK and PPP, and to ensure accurate seeding of the PPP engine from RTK, the regional datum must be provided with the `setGeodeticDatum` command.

Note that local RTK positions obtained after applying the datum transformation parameters transmitted in RTCM v3.x MT1021-1023 are never used for PPP seeding. In other words, the position reported in the `PosLocal` SBF block is not suitable for PPP seeding. See also section 1.3.4.5.

1.3.4.6.3 Tide Corrections

Since PPP is based on global satellite corrections, the PPP position would be sensitive to earth tide variations if no correction were applied. The receiver applies a tide correction based on the Sinko Earth tide model⁽⁴⁾. All positions reported in the `PVTCartesian`, `PVTGeodetic` and `PosCart` SBF blocks are always tide-corrected.

1.3.5 Receiver Autonomous Integrity Monitoring (RAIM)

The receiver features RAIM to ensure the integrity of the computed position solution, provided that sufficient satellites are available. The RAIM algorithm consists of three steps: detection, identification and adaptation, or shortly “D-I-A”⁽⁵⁾:

- Detection : an overall model statistical test is performed to assess whether an integrity problem has occurred;
- Identification : statistical w -tests are performed on each individual measurement to assess whether it should be marked as an outlier;
- Adaptation : measurements marked as an outlier are removed from the position computation to restore the integrity of the position solution. This step is only applied if outliers have been detected in the detection step.

⁽⁴⁾ Sinko, J., A Compact Earth Tides Algorithm for WADGPS. Proceedings of ION GPS-95, Palm Springs, California, September 12-15, 1995, pp. 35-44.

⁽⁵⁾ Baarda, W., A Testing Procedure For Use in Geodetic Networks, Netherlands Geodetic Commission, Publ. On Geodesy, Vol.2, no. 5, 1968

If the overall model statistical test fails, the RAIM module attempts to recover from the integrity failure by removing the responsible measurement(s) identified in the second step. As a consequence, the RAIM module will generally increase the continuity of integrity. The `IntegrityFlag` field of the `RAIMStatistics` SBF block reports an integrity failure if insufficient measurements remain after outlier removal (after several D-I-A steps), or if the overall model statistical test fails while no outliers can be identified. In the latter case the "sum of squared residuals too large" error is reported in the `Error` field of the PVT related SBF blocks.

The statistical tests assume an a-priori model of the measurement error probability distribution. As such, these tests can have the four classical outcomes in hypothesis testing, as shown in the table below (the letters A, B, C and D refer to the samples in Figure 1.3-4):

	<i>no outlier</i>	<i>outlier present</i>
<i>outlier detected</i>	False Alarm (type I error) A	Correct D
<i>no outlier detected</i>	Correct B	Missed Detection (type II error) C

The RAIM module makes a correct decision in two cases: an outlier present in the data is indeed detected, and no outlier is detected when none is present. However, when no outlier is present and the RAIM module declares an outlier is present, a false alarm is triggered. When an outlier remains undetected, a missed detection occurs.

The probability computations are based on the assumption that the residuals are distributed as a Normal distribution (central if there is no outlier, and non-central if there is one), as illustrated in Figure 1.3-4.

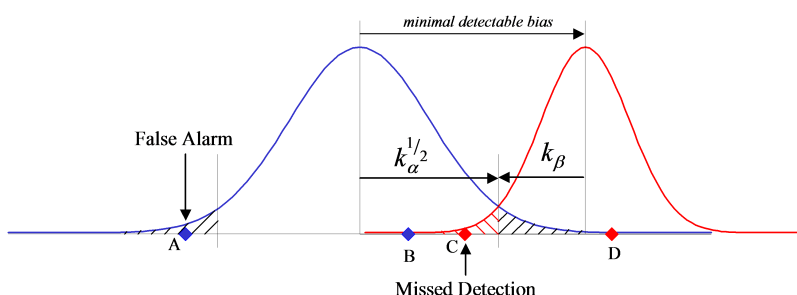


Figure 1.3-4: Statistical test outcomes.

Samples corresponding to the four test outcomes are represented in Figure 1.3-4: samples A and B are from the unbiased measurement distribution, while samples C and D are from a biased measurement distribution corresponding to an outlier. Since sample A is larger than the test threshold, it will be incorrectly flagged as an outlier (false alarm). Sample C is not detected as an outlier although it is part of the biased distribution (missed detection). The acceptable probability of false alarm and the probability of missed detection for the

application must be determined and provided to the receiver. This is the purpose of the **setRAIMLevels** command.

1.3.5.1 Integrity Algorithm

Two kinds of statistical tests are performed: the detection step uses an *overall model* test to evaluate the integrity of the position solution as a whole, and the identification step uses the *w*-test (also known as "datasnooping") to evaluate the integrity of individual measurements. Depending on the positioning mode, the overall model test is computed for range, range-rate and/or phase measurements simultaneously, while the *w*-test is computed for each range, range rate and/or phase measurement individually. Both the overall model and the *w*-tests are of the *Generalized Likelihood Ratio Test* type.

The overall model test uses the weighted sum of the squared residuals as test statistic. This test statistic is distributed as a χ^2 distribution with r degrees of freedom, where r is the redundancy number equal to the number of satellites used in the position computation minus 4. The test reads:

$$\sigma^2 = \bar{e}^T Q_y \bar{e} > \chi_\alpha^2(r, 0)$$

where:

- σ^2 is the overall model test statistic;
- \bar{e} is the vector of residuals;
- Q_y is the variance-covariance matrix of the measurements;
- $\chi_\alpha^2(r, 0)$ is the test threshold yielding a probability α of false alarm.

The probability of false alarm of the overall model test is selectable by the user with the *ModelReliability* argument of the **setRAIMLevels** command.

If the overall model test statistic is lower than the test threshold, the test is passed and the integrity is guaranteed under the statistical assumptions specified by the **setRAIMLevels** command.

If the overall model test statistic is higher than the threshold, the test is rejected. In this case, the identification step will attempt to identify the measurement responsible for the rejection using the *w*-test discussed below. After removal of the responsible outlier(s), the overall model test statistic is recomputed to verify the integrity of the solution without the outlier present. This iterative process continues until either the overall model test along with the associated *w*-tests are accepted, or until the *w*-tests for each individual measurement are accepted with a rejected overall model test. In the latter case an integrity loss is declared; in the former case integrity is available. Note that under extreme circumstances the interactive D-I-A process can also halt due to insufficient available measurements for testing, after removal of outliers. In this case the "too many outliers" error is reported in the PVT related SBF blocks.

For the evaluation of the *w*-test statistic, the following inequality is verified:

$$-k_\alpha^{1/2} < w_i = \frac{e_i}{\sigma_{e_i}} < +k_\alpha^{1/2}$$

where:

- w_i is the w -test statistic for the i th satellite;
- e_i is the residual for the i th satellite;
- σ_{e_i} is the standard deviation of the residual for the i th satellite;
- $k_\alpha^{1/2}$ is the test threshold yielding a probability α of false alarm.

The probability of false alarm of the w -test is selectable by the user with the Pfa argument of the **setRAIMLevels** command.

The test threshold is computed by the receiver with the assumption that the w -test statistic is distributed as a Normal distribution. For instance, if Pfa is set to 10%, residuals larger than 1.64 sigma are flagged as outliers. If Pfa is 0.01% the threshold will be 3.89.

1.3.5.2 Internal and External Reliability Levels

To assess the impact of undetected measurement errors on the computed position, the minimal detectable bias (MDB) in the range domain is computed and propagated to the position domain.

The MDB describes the internal reliability of the corresponding w -test. It is a measure of the range error that can be detected with a given probability of missed detection. It is computed as follows for each satellite (neglecting the probability that the biased measurement falls on the left-hand side of the non-biased distribution shown in Figure 1.3-4):

$$MDB_i = \sigma_{y_i} \left(\frac{\lambda_0}{1 - \frac{\sigma_{\hat{y}_i}^2}{\sigma_{y_i}^2}} \right)^{1/2}$$

where:

- σ_{y_i} is the standard deviation of the range measurement of the i th satellite;
- $\sigma_{\hat{y}_i}$ is the standard deviation of the estimator for the (measured) range of the i th satellite;
- λ_0 is the non-centrality parameter, which depends upon the probability of false alarm of the w -test and the probability of missed detection.

The user can select the probability of missed detection acceptable for his/her application with the Pmd argument of the **setRAIMLevels** command.

The external reliability is defined as the influence of a model error of size MDB on the user position. It is computed by propagating the MDB for each satellite to the position domain, taking the satellite geometry into account. The receiver computes a distinct external reliability level (XERL) for the horizontal and the vertical components (referred to as HERL and VERL respectively). These values should be compared to the alarm threshold of your specific application in order to verify if the position solution is adequate for that application.

Detailed results of the RAIM algorithm are available in the **RAIMStatistics** and the **PVT-Residual SBF** blocks and in the **GBS NMEA** message.

Chapter 2

Command Line Interface

2.1 Introduction

2.1.1 Scope

This document describes the ASCII command-line interface supported by your receiver. The Command and Log Reference Card provides a synopsis of all commands.

2.1.2 Typographical Conventions

abc	Commands to be entered by the user;
<i>abc</i>	Replies from the receiver;
<i>abc</i>	Command argument name.

2.2 Outline

The receiver outputs a prompt when it is ready to accept a user command. The prompt is of the form:

```
CD>
```

where **CD** is the connection descriptor of the current connection (e.g. **COMx** or **USBx**). For instance, if a user is connected to **COM1**, the prompt will be:

```
COM1>
```

Most commands fall into one of the following categories:

- set**-commands to change one or more configuration parameters;
- get**-commands to get the current value of one or more configuration parameters;
- exe**-commands to initiate some action;
- lst**-commands to retrieve the contents of internal files or list the commands.

Each **set**-command has its **get**-counterpart, but the opposite is not true. For instance, the **setNMEAOutput** command has a corresponding **getNMEAOutput**, but **getReceiverCapabilities** has no **set**-counterpart. Each **exe**-command also has its **get**-counterpart which can be used to retrieve the parameters of the last invocation of the command.

The prompt indicates the termination of the processing of a given command. When sending multiple commands to the receiver, it is necessary to wait for the prompt between each command.

2.2.1 Command Line Syntax

Each ASCII command line consists of a command name optionally followed by a list of arguments and terminated by <CR>, <LF> or <CR><LF> character(s) usually corresponding to pressing the "Enter" key on the keyboard.

To minimize typing effort when sending commands by hand, the command name can be replaced by its 3- or 4-character mnemonic. For instance, **grc** can be used instead of **getReceiverCapabilities**.

The receiver is case insensitive when interpreting a command line.

The maximum length of any ASCII command line is 2000 characters.

For commands requiring arguments, the comma "," must be used to separate the arguments from each other and from the command's name. Any number of spaces can be inserted before and after the comma.

Each argument of a **set**-command corresponds to a single configuration parameter in the receiver. Usually, each of these configuration parameters can be set independently of the others, so most of the **set**-command's arguments are optional. Optional arguments can be omitted but if omitted arguments are followed by non-omitted ones, a corresponding number of commas must be entered. Omitted arguments always keep their current value.

2.2.2 Command Replies

The reply to ASCII commands always starts with "\$R" and ends with <CR><LF> followed by the prompt corresponding to the connection descriptor you are connected to.

The following types of replies are defined for ASCII commands:

- For comment lines (user input beginning with "#") or empty commands (just pressing "Enter"), the receiver replies with the prompt.

```
COM1> # This is a comment! <CR>
COM1>
```

- For invalid commands, the reply is an error message, always beginning with the keyword "\$R?" followed by an error message. The different error messages are listed in Appendix A.
- For all valid **set**-, **get**- and **exe**-commands, the first line of the reply is an exact copy of the command as entered by the user, preceded with "\$R:". One or more additional lines are printed depending on the command. These lines report the configuration of the receiver after execution of the command.

```
COM1>setNMEAOutput, stream1, com1, GGA, sec1 <CR>
$R: setNMEAOutput, stream1, com1, GGA, sec1
    NMEAOutput, stream1, com1, GGA, sec1
COM1>
```

For commands which reset or halt the receiver (e.g. **exeResetReceiver**), the reply is terminated by "STOP>" instead of the standard prompt, to indicate that no further command can be entered.

- For all valid **lst**-commands, the first line of the reply is an exact copy of the command as entered by the user, preceded with "\$R;". The second line is a pseudo-prompt "---->" and the remaining of the reply is a succession of formatted blocks, each of them starting with "\$-- BLOCK".

ASCII replies to **set**-, **get**- and **exe**-commands, including the terminating prompt, are atomic: they cannot be broken by other messages from the receivers. For the **lst**-commands, the replies may consist of several atomic formatted blocks which can be interleaved with other output data. If more than one formatted block is output for a **lst**-command, each of the intermediate blocks is terminated with a pseudo-prompt "---->". The normal prompt will only be used to terminate the last formatted block of the reply so that one single prompt is always associated with one command.

2.2.3 Command Syntax Tables

All ASCII commands are listed in Section 2.3, "Command List". Each command is introduced by a compact formal description of it called a "syntax table". Syntax tables contain a complete list of arguments with their possible values and default settings when applicable.

The conventions used in syntax tables are explained below by taking a fictitious **setCommandName** command as example. The syntax table for that command is:

scn gcn	setCommandName getCommandName	Cd Cd	Distance	Time	Message (120)	Mode	PRN
		+Com1 +Com2 all	-20.00 ... <u>0.00</u> ... 20.00 m	<u>1</u> ... 50 sec	<u>Unknown</u>	<u>on</u> off	none +G01 ... G32 +S120 ... S138 +SBAS +GPS all

[RxControl: Navigation > Receiver Operation > Example](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Example](#)

The associated **set**- and **get**-commands are always described in pairs, and the same holds for the associated **exe**- and **get**-commands. The command name and its equivalent 3- or 4-character mnemonic are printed in the first two columns. The list of arguments for the **set**- and **get**commands is listed in the first and second row respectively. In our example, **setCommandName** can accept up to 6 arguments and **getCommandName** only accepts one argument. Mandatory arguments are printed in bold face. Besides the mandatory arguments, at least one of the optional arguments must be provided in the command line.

The list of possible values for each argument is printed under each of them. Default values for optional arguments are underlined.

The fictitious command above contains all the possible argument types:

- **Cd** serves as an index for all following arguments. This can be noticed by the possibility to use this argument in the **get**-command. This argument is mandatory in the **set**-command. The accepted values are **COM1**, **COM2** and **all**, corresponding to the first or second serial ports, or to both of them respectively. The "+" sign before the first two values indicates that they can be combined to address both serial ports in the same command.

Examples: **COM1**, **COM1+COM2**, **all** (which is actually an alias for **COM1+COM2**).

- **Distance** is a number between -20 and 20 with a default value of 0, and up to 2 decimal digits. An error is returned if more digits are provided. The "m" indicates that the value is expressed in meters. Note that this "m" should not be typed when entering the command.

Examples: 20, 10.3, -2.34

- **Time** is a number between 1 and 50, with no decimal digit (i.e. this is an integer value). This value is expressed in seconds.

Examples: 1, 10

- *Message* is a string with a maximum length of 120 characters. The default value of that argument is "Unknown". When spaces must to be used, the string has to be put between quotes and these enclosing quotes are not considered part of the string. The list of allowed characters in strings is:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
!#%&'()*+,-./:;<=>?[\]^_`{|}~
```

Example: "Hello World!"

- *Mode* is a range of individual values that cannot be combined (they are not preceded by a "+" sign). Either `off` or `on` can be selected for that argument and the default value is `on`.

Example: `on`

- *PRN* is a range of values that can be combined together with the "+" sign. The default value `GPS` is an alias for `G01+G02+ ... +G32`, `SBAS` is an alias for `S120+ ... +S138` and `all` an alias for `GPS+SBAS`. A "+" sign can be set before the argument to indicate to add the specified value(s) to the current list. If the value "none" is supported (which is the case in this example), a "-" sign can be set before the argument to remove the specified value(s) from the current list. It is possible to add or remove multiple values at once by "adding" or "subtracting" them with the "+" or "-" operator. However, "+" and "-" can never be combined in a single argument.

Examples: `G01+G02, +G03, GPS+S120, +G04+G05, -S122-S123, -GPS`

The lines printed in blue under the syntax table show under which menu the command can be found using RxControl or the Web Interface (the latter is only relevant for receivers supporting a web interface).

2.3 Command List

2.3.1 Receiver Administration Commands

lai	lstAntennaInfo	Antenna								
		Overview								
		Main								
		[antenna name]								

Use this command with the argument *Antenna* set to *Overview* to get a list of all antenna names for which the receiver knows the phase center variation parameters (see Firmware User Manual for a discussion on the phase center variation).

Use this command with the argument *Antenna* set to one of the antenna names returned by **lstAntennaInfo**, *Overview* to retrieve the complete phase center variation parameters for that particular antenna. Do not forget to enclose the name between double quotes if it contains whitespaces.

Using the values *Main* will return the phase center variation parameters corresponding to the main antenna type as specified in the command **setAntennaOffset**.

Examples

```
COM1> lai, Overview <CR>
$R; lai, Overview
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AntennaInfo version="0.1">
  <Antenna ID="AERAT1675_29      NONE"/>
  <Antenna ID="AERAT2775_150    NONE"/>
  <Antenna ID="AERAT2775_159    "/>
  <Antenna ID="AERAT2775_159    SPKE"/>
  <Antenna ID="AERAT2775_160    "/>
  ...
  <Antenna ID="TRM_R8_GNSS      "/>
</AntennaInfo>
COM1>

COM1> lai, "AERAT2775_159 SPKE" <CR>
$R; lai,"AERAT2775_159  SPKE"
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AntennaInfo version="0.1">
  <Antenna ID="AERAT2775_159  SPKE"/>
    <L1>
      <offset north="0.4" east="0.1" up="77.2"/>
      <phase elevation="90" value="0.0"/>
      <phase elevation="85" value="-0.2"/>
    ...
      <phase elevation=" 5" value="0.0"/>
      <phase elevation=" 0" value="0.0"/>
    </L2>
</AntennaInfo>
COM1>
```

help	1stCommandHelp	Action (255)								
		Overview								

Use this command to retrieve a short description of the ASCII command-line interface.

When invoking this command with the `Overview` argument, the receiver returns the list of all supported **set**-, **get**- and **exe**-commands. The `1stCommandHelp` command can also be called with any supported **set**-, **get**- or **exe**-command (the full name or the mnemonic) as argument.

The reply to this command is free-formatted and subject to change in future versions of the receiver's software. This command is designed to be used by human users. When building software applications, it is recommended to use the formal `1stMIBDescription`.

Examples

```
COM1> help, Overview <CR>
```

```
$R; help, Overview
```

```
$-- BLOCK 1 / 0
```

```
MENU: communication
```

```
GROUP: ioSelection
```

```
sdio, setDataInOut
```

```
gdio, getDataInOut
```

```
...
```

```
COM1>
```

```
COM1> help, getReceiverCapabilities <CR>
```

```
$R; help, getReceiverCapabilities
```

```
... Here comes a description of getReceiverCapabilities ...
```

```
COM1>
```

```
COM1> help, grc <CR>
```

```
$R; help, grc
```

```
... Here comes a description of getReceiverCapabilities ...
```

```
COM1>
```

lcf	IstConfigFile	File								
		Current								
		Boot								
		RxDDefault								
		User1								
		User2								

Use this command to list the contents of a configuration file. A configuration file contains the list of user commands needed to bring the receiver from factory default to a certain non-default configuration.

The following configuration files are available:

File	Description
Current	The current configuration.
Boot	The configuration that is loaded at boot time, after a power cycle or after a hard reset (see also the exeResetReceiver command).
RxDDefault	The default configuration.
User1	A user-defined configuration.
User2	A user-defined configuration.

See also the related **exeCopyConfigFile** command to learn how to manage configuration files.

Example

```
COM1> smp, TestMarker <CR>
$R: smp, TestMarker
    MarkerParameters, "TestMarker"
COM1> lcf, Current <CR>
$R; lcf, Current
$-- BLOCK 1 / 1
    setMarkerParameters, "TestMarker"
COM1>
```

eccf gccf	exeCopyConfigFile getCopyConfigFile	Source	Target							
		Current	Current							
		Boot	Boot							
		User1	User1							
		User2	User2							
		RxDefault								

RxControl: File > Copy Configuration

Web Interface: Receiver > Administration > Copy Configuration

Use this command to manage the configuration files. See the **1stConfigFile** command for a description of the different configuration files.

With this command, the user can copy configurations files into other configuration files. For instance, copying the `Current` file into the `Boot` file makes that the receiver will always boot in the current configuration.

Examples

To save the current configuration in the `Boot` file, use:

```
COM1> eccf, Current, Boot <CR>
$R: eccf, Current, Boot
    CopyConfigFile, Current, Boot
COM1>
```

To load the configuration stored in `User1`, use:

```
COM1> eccf, User1, Current <CR>
$R: eccf, User1, Current
    CopyConfigFile, User1, Current
COM1>
```

efup gfup	exeFTPUpgrade getFTPUpgrade	Server (32)	Path (64)	Login (12)	Password (24)					
				anonymous						

RxControl: File > Upgrade Receiver using FTP

Web Interface: Receiver > Administration > Upgrade Receiver using FTP

Use this command to upgrade the receiver by fetching the upgrade file from an FTP server. The arguments specify the FTP server, the path to the upgrade file (.SUF format), and the login and password to use.

This procedure always resets the receiver, even if the upgrade file does not exist.

Before resetting, the receiver broadcasts a "\$TE ResetReceiver" message to all active communication ports, to inform all users of the imminent reset.

After a reset, the user may have to adapt the communication settings of his/her terminal program as they may be reset to their default values.

Example

```
COM1> efup, myftp.com, /tst.suf, user, password<CR>
$R: efup, myftp.com, /tst.suf, user, password
    FTPUpgrade, "myftp.com", "/tst.suf", "user", "
        I301I5B8DG8E7QTT6RZT7IQ"
STOP>
$TE ResetReceiver Upgrade
STOP>
```

lif	lstInternalFile	File								
		Permissions Identification Debug Error SisError DiffCorrError SetupError LBAS1Access LBAS1Subscr IPParameters								

Use this command to retrieve the contents of one of the receiver internal files:

File	Description
Permissions	List of permitted options in your receiver.
Identification	Information about the different components being part of the receiver (e.g. serial number, firmware version, etc.).
Debug	Program flow information that can help support engineers to debug certain issues.
Error	Last internal error reports.
SisError	Last detected signal-in-space anomalies.
DiffCorrError	Last detected anomalies in the incoming differential correction streams.
SetupError	Last detected anomalies in the receiver setup.
LBAS1Access	LBAS1 (L-Band Augmentation Service 1) Access information.
LBAS1Subscr	LBAS1 (L-Band Augmentation Service 1) Subscription status.
IPParameters	Hostname, MAC and IP addresses, DNS addresses, netmask and gateway (gateway shown only in static IP mode, see command setIPSettings).

Example

```
COM1> lif, Permissions <CR>
$R; lif, Permissions
---->
$-- BLOCK 1 / 1
... here follows the permission file ...
COM1>
```


lmd	1stMIBDescription	File (255)								
		Overview SBFTable								

Use this command to retrieve the ASN.1-compliant syntax of the user command interface. The name of the command refers to the MIB (Management Information Base), which holds the whole receiver configuration. There is a one-to-one relationship between the formal MIB description and the ASCII command-line interface for all **exe**-, **get**- and **set**-commands.

When the value `Overview` is used, the general syntax of the interface is returned. With the value `SBFTable`, the receiver will output the list of supported SBF blocks and whether they can be output at a user-selectable rate or not. The **1stMIBDescription** command can also be called with every supported **set**-, **get**- or **exe**-command (the full name or the mnemonic) as argument.

No formal description of the **1st**-commands can be retrieved with **1stMIBDescription**.

Examples

```
COM1> lmd, Overview <CR>
$R; lmd, Overview
... Here comes the generic command syntax ...
COM1>
```

```
COM1> lmd, grc <CR>
$R; lmd, grc
... Here comes the description of getReceiverCapabilities ...
COM1>
```

epwm gpwm	exePowerMode getPowerMode	Mode								
		ScheduledSleep								
		StandBy								

RxControl: File > Power Mode > Shut Down

Web Interface: Receiver > Administration > Power Mode > Shut Down

Use this command to set the receiver in sleep or stand-by mode, in which it consumes only a fraction of its normal operational power.

When in `ScheduledSleep` or `StandBy` mode, the receiver can be awoken by sending the appropriate signal to one of its input pins, or by sending a character to the first COM port (see the Hardware Manual for details).

When in `ScheduledSleep` mode, the receiver can also automatically wake up at a given time or at regular intervals. This functionality is controlled by the `setWakeUpInterval` command.

Upon waking up, the receiver applies the configuration that is stored in the boot configuration file (see the `1stConfigFile` command).

Before entering sleep or stand-by mode, the receiver broadcasts a "\$TE PowerMode" message to all active communication ports, to inform all users of the imminent halt.

Example

```
COM1> epwm, ScheduledSleep <CR>
$R: epwm, ScheduledSleep
    PowerMode, ScheduledSleep
STOP>
$TE PowerMode ScheduledSleep
STOP>
```

grc	getReceiverCapabilities									

[RxControl: Help > Receiver Interface > Permitted Capabilities](#)

[Web Interface: Configuration > Help > Receiver Interface > Permitted Capabilities](#)

Use this command to retrieve the so-called "capabilities" of your receiver. The first returned value is the list of supported antenna(s), followed by the list of supported signals, the list of available communication ports and the list of enabled features.

The three values at the end of the reply line correspond to the default measurement interval, the default PVT interval and the default integrated INS/GNSS interval respectively. This is the interval at which the corresponding SBF blocks are output when the `OnChange` rate is selected with the `setSBFOutput` command. These values are expressed in milliseconds.

Each of the above-mentioned lists contain one or more of the elements in the tables below.

Antennas	Description
Main	The receiver's main antenna.

Signals	Description
GPSL1CA	GPS L1 C/A signal.
GPSL1PY	GPS L1 P(Y) signal.
GPSL2PY	GPS L2 P(Y) signal.
GPSL2C	GPS L2 C signal.
GPSL5	GPS L5 signal.
GLOL1CA	GLONASS L1 C/A signal.
GLOL2P	GLONASS L2 P signal.
GLOL2CA	GLONASS L2 C/A signal.
GLOL3	GLONASS L3 signal.
GALL1BC	Galileo L1 BC signal.
GALE5a	Galileo E5a signal.
GALE5b	Galileo E5b signal.
GALE5	Galileo E5 AltBOC signal.
GEOL1	SBAS L1 C/A signal.
GEOL5	SBAS L5 signal.
CMPL1	COMPASS/BEIDOU B1 signal.
CMPE5b	COMPASS/BEIDOU B2 signal.
QZSL1CA	QZSS L1 C/A signal.
QZSL2C	QZSS L2 C signal.
QZSL5	QZSS L5 signal.
LBAND	MSS L-Band signal.

ComPorts	Description
COM1	Serial port 1.
COM2	Serial port 2.
COM3	Serial port 3.
COM4	Serial port 4.
USB1	Virtual serial port 1.
USB2	Virtual serial port 2.
IP10	TCP/IP port 1.

ComPorts (Continued)	Description
IP11	TCP/IP port 2.
IP12	TCP/IP port 3.
IP13	TCP/IP port 4.
IP14	TCP/IP port 5.
IP15	TCP/IP port 6.
IP16	TCP/IP port 7.
IP17	TCP/IP port 8.
NTR1	NTRIP port 1.
NTR2	NTRIP port 2.
NTR3	NTRIP port 3.
IPS1	IP Server port 1.
IPS2	IP Server port 2.
IPS3	IP Server port 3.
IPR1	IP Receive port 1.
IPR2	IP Receive port 2.
IPR3	IP Receive port 3.

Capabilities	Description
SBAS	Positioning with SBAS corrections.
DGPSRover	Positioning with DGPS corrections.
DGPSBase	Generation of DGPS corrections.
RTKRover	Positioning with RTK corrections.
RTKBase	Generation of RTK corrections.
RTCMv23	Generation/decoding of RTCM v2.3 corrections.
RTCMv3x	Generation/decoding of RTCM v3.x corrections.
CMRv20	Generation/decoding of CMR v2.0 corrections.
xPPSInput	Internal clock synchronisation with xPPS input signal.
xPPSOutput	Generation of xPPS output signal.
TimedEvent	Accurate time mark of event signals.
InternalLogging	Internal logging.
APME	A-Posteriori Multipath Estimator.
RAIM	Receiver Autonomous Integrity Monitoring.
PPPGlobal	PPP through Wide Area Augmentation Service for global use.
PPPLand	PPP through Wide Area Augmentation Service for land based use.
LBAS1	L Band Augmentation data Service 1.
LBAS1L	L Band Augmentation data Service 1 Land only.
SIGIL	Usage of SIGIL input allowed.
MovingBase	Usage of Moving Base allowed.

Example

```
COM1> grc <CR>
$R: grc
    ReceiverCapabilities, Main, GPSL1CA+GEOL1, COM1+COM2+COM3+COM4+
        USB1+USB2,
APME+SBAS, 100, 100, 100
COM1>
```

gri	getReceiverInterface	Item								
		+ RxName + SNMPLanguage + SNMPVersion all								

[RxControl: Help > Receiver Interface > Interface Version](#)

[Web Interface: Configuration > Help > Receiver Interface > Interface Version](#)

Use this command to retrieve the version of the receiver command-line interface. The reply to this command is a subset of the reply returned by the **1stInternalFile**, **Identification** command.

Example

```
COM1> gri <CR>
$R: gri
ReceiverInterface, RxName, AsteRx1
ReceiverInterface, SNMPLanguage, English
ReceiverInterface, SNMPVersion, 20060308
COM1>
```

era gra	exeRegisteredApplications getRegisteredApplications	Cd Cd	Application (12)							
		+ COM1 + COM2 + COM3 + COM4 + USB1 + USB2 + IP10 ... IP17 all	Unknown							

[RxControl: Communication > Registration](#)

[Web Interface: Configuration > Communication > Registration](#)

Use these commands to define/inquire the name of the application that is currently using a given connection descriptor (*Cd*).

Registering an application name for a connection does not affect the receiver operation, and is done on a voluntary basis. Application registration can be useful to developers of external applications when more than one application is to communicate with the receiver concurrently. Whether or not this command is used, and the way it is used is up to the developers of external applications.

Example

```
COM1> era, com1, MyApp <CR>
$R: era, com1, MyApp
RegisteredApplications, COM1, "MyApp"
RegisteredApplications, COM2, "Unknown"
RegisteredApplications, COM3, "Unknown"
RegisteredApplications, USB1, "Unknown"
RegisteredApplications, USB2, "Unknown"
COM1>
```

erst grst	exeResetReceiver getResetReceiver	Level	EraseMemory							
		Soft Hard Upgrade	none + Config + PVTData + SatData + BaseStations all							

RxControl: File > Reset Receiver

Web Interface: Receiver > Administration > Reset Receiver

Use this command to reset the receiver and to erase some previously stored data. The first argument specifies which level of reset you want to execute:

Level	Description
Soft	This is a reset of the receiver's firmware. After a few seconds, the receiver will restart operating in the same configuration as before the command was issued, unless the "Config" value is specified in the second argument.
Hard	This is similar to a power off/on sequence. After hardware reset, the receiver will use the configuration saved in the boot configuration file.
Upgrade	Set the receiver into upgrade mode. After a few seconds, the receiver is ready to accept an upgrade file (SUF format) from any of its connections.

The second argument specifies which part of the non-volatile memory should be erased during the reset. The following table contains the possible values for the *EraseMemory* argument:

EraseMemory	Description
Config	The receiver's configuration is reset to the factory default. The Current and Boot configuration files are erased (see the exeCopyConfigFile command). Note that the User1 and User2 configuration files are not erased: use the exeCopyConfigFile command instead. Also, the IP settings set by the commands setIPSettings and setIPPortSettings are not reset.
PVTData	The latest computed PVT data stored in non-volatile memory is erased.
SatData	All satellite navigation data (ephemeris, almanac, ionosphere parameters, UTC, ...) stored in non-volatile memory is erased.
BaseStations	All base stations stored in non-volatile memory are erased.

Before resetting, the receiver broadcasts a "\$TE ResetReceiver" message to all active communication ports, to inform all users of the imminent reset.

After a reset, the user may have to adapt the communication settings of his/her terminal program as they may be reset to their default values.

Example

```
COM1> erst, soft, none <CR>
$R: erst, soft, none
```



```
ResetReceiver, Soft, none  
STOP>  
$TE ResetReceiver Soft  
STOP>
```

2.3.2 Authentication Commands

lcu	lstCurrentUser									

Use this command to check which user is currently logged in on this port, if any. See also the **login** command.

Example

```
COM1> lcu <CR>
$R! lstCurrentUser
    Not logged in.
COM1> login, admin, admin <CR>
$R! LogIn
    User admin logged in.
COM1> lcu <CR>
$R! lstCurrentUser
    Logged in as admin.
COM1>
```

sdal gdal	setDefaultAccessLevel getDefaultAccessLevel	Web	Ftp	Ip	Com	Usb				
		none Viewer User	none Viewer User	none Viewer User	none Viewer User	none Viewer User				

RxControl: File > User Management

Web Interface: Receiver > Administration > User Management

This command defines what an anonymous user is authorized to do when connected to the receiver. An anonymous user is one who has not logged in with the **login** command.

The anonymous authorization level can be set independently for the Web interface, the FTP access, TCP/IP ports, COM ports and USB ports.

For the *Web*, *Ip*, *Com* and *Usb* arguments, setting the authorization level to **User** grants full control of the receiver to the anonymous user connected through that connection. The **Viewer** level allows the anonymous user to view the receiver configuration without changing it (i.e. to only issue **get**-commands). **none** prevents anonymous users from viewing or changing the configuration.

For the *Ftp* argument, **Viewer** means that the anonymous user is allowed to download files, but not to delete them. **User** means that the anonymous user can both download and delete files.

To perform actions not allowed to anonymous users, you first need to authenticate yourself by entering a *UserName* and *Password* through the **login** command.

See also the commands **setUserAccessLevel** to learn how to define user accounts.

This command does not change the status of existing connections. In particular, for *Com* and *Usb* connections, it will only take effect after a reset.

Example

To require logging in on Web and FTP interfaces, use:

```
COM1> sdal, none, none <CR>
$R: sdal, none, none
    DefaultAccessLevel, none, none, User, User, User
COM1>
```

login	Login	UserName (16)	Password (32)							

Use this command to authenticate yourself. When initially connecting to the receiver, a user is considered "anonymous". The level of control granted to anonymous users is defined by the command **setDefaultAccessLevel**.

To perform actions not allowed to anonymous users, you need to authenticate yourself by entering a *UserName* and *Password* through the **login** command.

The list of user names and passwords and their respective access level can be managed with the **setUserAccessLevel** command. Login fails if the provided *UserName* or *Password* is not in that list.

The **logout** command returns to unauthenticated (anonymous) access. The **lstCurrentUser** command can be invoked to find out which user is logged in on the current port.

It is not necessary to log out before logging in as a different user.

Examples

To log in as user "admin" with password "admin", use

```
COM1> login, admin, admin <CR>
$R! LogIn
    User admin logged in.
COM1>
```

Logging in with a wrong username or password gives an error:

```
COM1> login, foo, foo <CR>
$R? LogIn: Wrong username or password!
COM1>
```

If the user does not have sufficient access right, some commands may give an error:

```
COM1> sso, Stream1, COM1, MeasEpoch, sec1 <CR>
$R? SBFOutput: Not authorized!
COM1>
```

logout	LogOut									

Use this command to return to anonymous access. It is the reverse of **login**.

Example

The following sequence of commands logs in as user "admin" with password "admin", reconfigures SBF output, and logs out again:

```
COM1> login, admin, admin <CR>
$R! LogIn
    User admin logged in.
COM1> sso, Stream1, COM1, Group1, sec1 <CR>
$R: sso, Stream1, COM1, Group1, sec1
    SBFOutput, Stream1, COM1, Group1, sec1
COM1> logout <CR>
$R! LogOut
    User admin logged out.
COM1>
```

sual	setUserAccessLevel	UserID	UserName (16)	Password (32)	UserLevel					
gual	getUserAccessLevel	UserID								
		+ User1 ... User8			Viewer					
		all			User					

[RxControl: File > User Management](#)

[Web Interface: Receiver > Administration > User Management](#)

Use these commands to manage the users and their access rights on the receiver. Up to eight users can be defined (User1 to User8).

Each user is identified with a *UserName* and *Password*, and has a certain level of access (*UserLevel*). If *UserLevel* is `User`, the user has full control of the receiver. If it is `Viewer`, the user can only issue `get`-commands.

Note that the receiver encrypts the password so that it cannot be read back with the command `getUserAccessLevel`.

Examples

To create a user with name `logger`, password `xghNF%d` and "Viewer" access permissions:

```
COM1> sual, User3, logger, xghNF%d, Viewer <CR>
$R: sual, User3, logger, xghNF%d, Viewer
    UserAccessLevel, User3, "logger", "15872ICUR219W7S7ZL3543B5ECU",
    Viewer
COM1>
```

To remove a user from the list, use the empty string "" as *UserName* and *Password*:

```
COM1> sual, User3, "", "", none <CR>
$R: sual, User3, "", "", none
    UserAccessLevel, User3, "", "", none
COM1>
```

2.3.3 Tracking Configuration Commands

sam gam	setAGCMode getAGCMode	Band Band	Mode	Gain						
		+ L1 + L2L5 all	auto frozen manual	0 ... 35 ... 70 dB						

RxControl: Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Web Interface: Configuration > Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Use these commands to define/inquire the operation mode of the Automatic Gain Control (AGC) in the receiver frontend. The AGC is responsible for amplifying the input RF signal to an appropriate level.

By default (*Mode* is set to `auto`), the AGC automatically adjusts its gain in function of the input signal power. In `frozen` mode, the AGC gain is kept constant at its current value (after a ten-second stabilisation period) and does not follow any subsequent variation of the input signal power. In `manual` mode, the user can set the gain to a fixed value specified by the *Gain* argument. The *Gain* argument is ignored in `auto` and `frozen` modes.

The first argument (*Band*) specifies for which frequency band the settings apply.

Example

```
COM1> sam, all, frozen <CR>
$R: sam, all, frozen
      AGCMode, L1, frozen, 30
COM1>
```


sca gca	setChannelAllocation getChannelAllocation	Channel Channel	Satellite	Search	Doppler	Window				
		+ Ch01 ... Ch40 all	auto G01 ... G32 F01 ... F14 E01 ... E32 S120 ... S158 C01 ... C37 J01 J02 J03	auto manual	-50000 ... 0 ... 50000 Hz	1 ... 16000 ... 100000 Hz				

RxControl: [Navigation > Advanced User Settings > Channel Allocation](#)

Web Interface: [Configuration > Navigation > Advanced User Settings > Channel Allocation](#)

Use these commands to define/inquire the satellite-to-channel allocation of the receiver.

The action of the **setChannelAllocation** command is to force the allocation of a particular satellite on the set of channels identified with the *Channel* argument, thereby overruling the automatic channel allocation performed by the receiver. It is possible to allocate the same satellite to more than one channel. If you assign a satellite to a given channel, any other channel that was automatically allocated to the same satellite will be stopped and will be reallocated.

The values *Gxx*, *Exx*, *Fxx*, *Sxxx*, *Cxx* and *Jxx* for the *Satellite* argument represent GPS, Galileo, GLONASS, SBAS, Compass/BeiDou and QZSS satellites respectively. For GLONASS, the frequency number (with an offset of 8) should be provided, and not the slot number (hence the "F"). Setting the *Satellite* argument to *auto* brings the channel back in auto-allocation mode.

The user can specify the Doppler window in which the receiver has to search for the satellite. This is done by setting the *Search* argument to *manual*. In that case, the *Doppler* and *Window* arguments can be provided: the receiver will search for the signal within an interval of *Window* Hz centred on *Doppler* Hz. The value to be provided in the *Doppler* argument is the expected Doppler at the GPS L1 carrier frequency (1575.42MHz). This value includes the geometric Doppler and the receiver and satellite frequency biases. Specifying a Doppler window can speed up the search process in some circumstances. A satellite already in tracking that falls outside of the prescribed window will remain in tracking.

If *Search* is set to *auto*, the receiver applies its usual search procedure, as it would do for auto-allocated satellites, and the *Doppler* and *Window* arguments are ignored.

Be aware that this command may disturb the normal operation of the receiver and is intended only for expert-level users.

Examples

```
COM1> sca, Ch05, G01 <CR>
$R: sca, Ch05, G01
    ChannelAllocation, Ch05, G01, auto, 0, 16000
COM1>
```

```
COM1> gca, Ch05 <CR>
$R: gca, Ch05
    ChannelAllocation, Ch05, G01, auto, 0, 16000
COM1>
```

gcc	getChannelConfiguration	Channel								
		+Ch01 ... Ch40 all								

RxControl: Navigation > Advanced User Settings > Channel Configuration

Web Interface: Configuration > Navigation > Advanced User Settings > Channel Configuration

Use this command to get the list of signals that a given channel can track.

Example

To display the different signals that the first channel can track, use:

```
COM1> gcc, Ch01 <CR>
$R: gcc, Ch01
      ChannelConfiguration, Ch01, GPSL1CA
COM1>
```

scm gcm	setCN0Mask getCN0Mask	Signal Signal	Mask							
		+GPSL1CA +Reserved1 +Reserved2 +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +GEOL1 +GEOL5 +CMPL1 +CMPE5b +QZSL1CA +QZSL2C +QZSL5 all	0 ... 10 ... 60 dB-Hz							

RxControl: Navigation > Receiver Operation > Masks

Web Interface: Configuration > Navigation > Receiver Operation > Masks

Use these commands to define/inquire the carrier-to-noise ratio mask for the generation of measurements. The receiver does not generate measurements for those signals of which the C/N_0 is under the specified mask, and does not include these signals in the PVT computation. However, it continues to track these signals and to decode and use the navigation data as long as possible, regardless of the C/N_0 mask.

The mask can be set independently for each of the signal types supported by the receiver, except for the GPS P-code, of which the mask is fixed at 1 dB-Hz (this is because of the codeless tracking scheme needed for GPS P-code).

Examples

```
COM1> scm, GEOL1, 30 <CR>
$R: scm, GEOL1, 30
    CN0Mask, GEOL1, 30
COM1>
```

```
COM1> gcm, GEOL1 <CR>
$R: gcm, GEOL1
    CN0Mask, GEOL1, 30
COM1>
```

sfm	setFrontendMode	Mode								
gfm	getFrontendMode									
		Nominal								
		GLOL2Blocked								

RxControl: Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Web Interface: Configuration > Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Use these commands to define/inquire the frontend operating mode. The following modes are available.

Mode	Description
Nominal	Nominal operation.
GLOL2Blocked	GLONASS L2 band blocked. This mode should only be selected in case of a strong interferer in the GLONASS L2 band. In GLOL2Blocked mode, GLONASS L2 cannot be tracked.

Example

```
COM1> sfm, Nominal <CR>
$R: sfm, Nominal
      FrontendMode, Nominal
COM1>
```

smm	setMultipathMitigation	Code	Carrier							
gmm	getMultipathMitigation									
		off	off							
		on	on							

[RxControl: Navigation > Receiver Operation > Tracking and Measurements > Multipath](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Tracking and Measurements > Multipath](#)

Use these commands to define/inquire whether multipath mitigation is enabled or not.

The arguments *Code* and *Carrier* enable or disable the A-Posteriori Multipath Estimator (APME) for the code and carrier phase measurements respectively. APME is a technique by which the receiver continuously estimates the multipath error and corrects the measurements accordingly.

This multipath estimation process slightly increases the thermal noise on the pseudoranges. However, this increase is more than compensated by the dramatic decrease of the multipath noise.

Examples

```
COM1> smm, on, off <CR>
$R: smm, on, off
    MultipathMitigation, on, off
COM1>
```

```
COM1> gmm <CR>
$R: gmm
    MultipathMitigation, on, off
COM1>
```

snf gnf	setNotchFiltering getNotchFiltering	Notch Notch	Mode	CenterFreq	Bandwidth					
		+ Notch1 all	auto off manual	1100.000 ... 1700.000 MHz	30 ... 1600 kHz					

RxControl: Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Web Interface: Configuration > Navigation > Advanced User Settings > Frontend and Interference Mitigation > Frontend Settings

Use these commands to set the position of the notch filter(s) in the receiver's frontend. Notch filters are used to cancel narrowband interferences.

The *Mode* argument is used to enable or disable the notch filter specified in the first argument. When set to *auto*, the receiver performs automatic detection of the region of the spectrum affected by interference if any. In *manual* mode, the user forces a certain region of the spectrum to be blanked by the notch filter. That region must be specified by the arguments *CenterFreq* and *Bandwidth*. *Bandwidth* is the double-sided bandwidth centered at *CenterFreq*. Specifying a region outside of a GNSS band has no effect.

Example

```
COM1> snf, Notch1, manual, 1227, 30 <CR>
$R: snf, Notch1, manual, 1227, 30
    NotchFiltering, Notch1, manual, 1227.000, 30
COM1>
```

sst gst	setSatelliteTracking getSatelliteTracking	Satellite								
		none +G01 ... G32 +R01 ... R30 +E01 ... E32 +S120 ... S158 +C01 ... C37 +J01 +J02 +J03 +GPS +GLONASS +GALILEO +SBAS +COMPASS +QZSS all								

RxControl: Navigation > Advanced User Settings > Tracking > Satellite Tracking

Web Interface: Configuration > Navigation > Advanced User Settings > Tracking > Satellite Tracking

Use these commands to define/inquire which satellites are allowed to be tracked by the receiver. It is possible to enable or disable a single satellite (e.g. G01 for GPS PRN1), or a whole constellation. Gxx, Exx, Rxx, Sxxx, Cxx and Jxx refer to a GPS, Galileo, GLONASS, SBAS, Compass/BeiDou or QZSS satellite respectively. GLONASS satellites must be referenced by their slot number in this command.

For a satellite to be effectively tracked by the receiver, make sure that at least one of its signals is enabled in the **setSignalTracking** command.

A satellite which is disabled by this command is not considered anymore in the automatic channel allocation mechanism, but it can still be forced to a given channel, and tracked, using the **setChannelAllocation** command.

Tracking a satellite does not automatically mean that the satellite will be included in the PVT computation. The inclusion of a satellite in the PVT computation is controlled by the **setSatelliteUsage** command.

Examples

To only enable the tracking of GPS satellites, use:

```
COM1> sst, GPS <CR>
$R: sst, GPS
  SatelliteTracking, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26+G27
+G28+G29+G30+G31+G32
COM1>
```

To add all SBAS satellites in the list of satellites to be tracked, use:

```
COM1> sst, +SBAS <CR>
$R: sst, +SBAS
  SatelliteTracking, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26+G27
...
COM1>
```


To remove SBAS PRN120 from the list of allowed satellites, use:

```
COM1> sst, -S120 <CR>
```

```
$R: sst, -S120
```

```
    SatelliteTracking, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11  
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26+G27
```

```
...
```

```
    COM1>
```

snt gnt	setSignalTracking getSignalTracking	Signal								
		+GPSL1CA +GPSL1PY +GPSL2PY +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +GEOL1 +GEOL5 +CMPL1 +CMPE5b +QZSL1CA +QZSL2C +QZSL5 all								

RxControl: [Navigation > Advanced User Settings > Tracking > Signal Tracking](#)

Web Interface: [Configuration > Navigation > Advanced User Settings > Tracking > Signal Tracking](#)

Use these commands to define/inquire which signals are allowed to be tracked by the receiver.

Beware that disabling a given signal can prevent the receiver from tracking other signals: disabling GPSL1CA prevents the tracking of GPSL1PY, GPSL2PY and GPSL2C, disabling QZSL1CA prevents the tracking of QZSL2C, and disabling GLOL2CA prevents the tracking of GLOL2P.

Examples

To configure the receiver in a single-frequency L1 GPS+SBAS mode, use:

```
COM1> snt, GPSL1CA+GEOL1 <CR>
$R: snt, GPSL1CA+GEOL1
      SignalTracking, GPSL1CA+GEOL1
COM1>
```

```
COM1> gnt <CR>
$R: gnt
      SignalTracking, GPSL1CA+GEOL1
COM1>
```

ssi gsi	setSmoothingInterval getSmoothingInterval	Signal Signal	Interval	Alignment						
		+GPSL1CA +GPSL2PY +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +GEOL1 +GEOL5 +CMPL1 +CMPE5b +QZSL1CA +QZSL2C +QZSL5 all	0 ... 1000 sec	0 ... 1000 sec						

[RxControl: Navigation > Receiver Operation > Tracking and Measurements > Smoothing](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Tracking and Measurements > Smoothing](#)

Use these commands to define/inquire the code measurement smoothing interval.

The *Interval* argument defines the length of the smoothing filter that is used to smooth the code measurements by the carrier phase measurements. It is possible to define a different interval for each signal type. If *Interval* is set to 0, the code measurements are not smoothed. The smoothing interval can vary from 1 to 1000 seconds.

To prevent transient effect to perturb the smoothing filter, smoothing is disabled during the first ten seconds of tracking, i.e. when the lock time is lower than 10s. Likewise, the smoothing effectively starts with a delay of 10 seconds after entering the **setSmoothingInterval** command.

Code smoothing allows reducing the pseudoranges noise and multipath. It has no influence on the carrier phase and Doppler measurements. The smoothing filter has an incremental effect; the noise of the filtered pseudoranges will decrease over time and reach its minimum after *Interval* seconds. For some applications, it may be necessary to wait until this transient effect is over before including the measurement in the PVT computation. This is the purpose of the *Alignment* argument. If *Alignment* is not set to 0, measurements taken during the first *Alignment*+10 seconds of tracking will be discarded. The effective amount of *Alignment* is never larger than *Interval*, even if the user sets it to a larger value.

Examples

```
COM1> ssi, GPSL1CA, 300 <CR>
$R: ssi, GPSL1CA, 300
    SmoothingInterval, GPSL1CA, 300, 0
COM1>
```

```
COM1> gsi, GPSL1CA <CR>
$R: gsi, GPSL1CA
    SmoothingInterval, GPSL1CA, 300, 0
COM1>
```

stlp gtlp	setTrackingLoopParameters getTrackingLoopParameters	Signal Signal	DLLBandwidth	PLLBandwidth	MaxTpDLL	MaxTpPLL	Adaptive			
		+ GPSL1CA + Reserved1 + Reserved2 + GPSL2C + GPSL5 + GLOL1CA + GLOL2P + GLOL2CA + GLOL3 + GALL1BC + GALE5a + GALE5b + GALE5 + GEOL1 + GEOL5 + CMPL1 + CMPE5b + QZSL1CA + QZSL2C + QZSL5 all	0.01 ... 0.25 ... 5.00 Hz	1 ... 15 ... 100 Hz	1 ... 100 ... 500 msec	1 ... 10 ... 200 msec	off on			

RxControl: [Navigation > Advanced User Settings > Tracking > Tracking Loop Parameters](#)

Web Interface: [Configuration > Navigation > Advanced User Settings > Tracking > Tracking Loop Parameters](#)

Use these commands to define/inquire the tracking loop parameters for each individual signal type.

The *DLLBandwidth* and *PLLBandwidth* arguments define the single-sided DLL and PLL noise bandwidth, in Hz.

The *MaxTpDLL* argument defines the maximum DLL pre-detection time, in millisecond. The actual pre-detection time applied by the receiver (*TpDLL*) depends on the presence of a pilot component. For signals having a pilot component (e.g. GPS L2C), *TpDLL* = *MaxTpDLL*. For signals without pilot component (e.g. GPS L1CA), *TpDLL* is the largest divider of the symbol duration smaller than or equal to *MaxTpDLL*.

The *MaxTpPLL* argument defines the maximal PLL pre-detection time, in millisecond. The actual pre-detection time in the receiver (*TpPLL*) is computed in the same way as indicated for the *MaxTpDLL* argument.

Setting the *Adaptive* argument to `on` allows the receiver to dynamically change the loop parameters in order to optimize performance in specific conditions.

After entering this command, all active tracking loops stop and restart with the new settings.

This command should only be used by expert users who understand the consequences of modifying the default values. In some circumstances, changing the tracking parameters may result in the impossibility for the receiver to track a specific signal, or may significantly increase the processor load. It is recommended that the product of *TpPLL* (in milliseconds) and *PLLBandwidth* (in Hz) be kept between 100 and 200.

Note that decreasing the predetection times increases the load on the processor.

Example

```
COM1> stlp, GPSL1CA, 0.20, 12 <CR>
$R: stlp, GPSL1CA, 0.20, 12
    TrackingLoopParameters, GPSL1CA, 0.20, 12, 100, 10
```

COM1>

2.3.4 Navigation Configuration Commands

sal gal	setAntennaLocation getAntennaLocation	Antenna Antenna	Mode	DeltaX	DeltaY	DeltaZ				
		+ Base	auto	-1000.0000	-1000.0000	-1000.0000				
		all	manual	... 0.0000	... 0.0000	... 0.0000				
				... 1000.0000 m	... 1000.0000 m	... 1000.0000 m				

RxControl: Navigation > Positioning Mode > GNSS Attitude

Web Interface: Configuration > Navigation > Positioning Mode > GNSS Attitude

Use this command to define/inquire the relative location of the antennas in the vehicle reference frame in the context of attitude determination.

The attitude of a vehicle (more precisely the heading and pitch angles) can be determined from the orientation of the baseline between two antennas attached to the vehicle. These two antennas must be connected to two receivers configured in moving-base RTK operation. Use the command **setGNSSAttitude** to enable moving-base attitude determination.

The **setAntennaLocation** command should be invoked at the rover receiver with the *Antenna* argument set to `Base` to specify the relative position of the base antenna with respect to the rover antenna.

In `auto` mode, the receiver determines the attitude angles assuming that the baseline between the antenna ARPs is parallel to the longitudinal axis of the vehicle, and that the base antenna is in front of the rover antenna (i.e. towards the direction of movement). The length of the baseline is automatically computed by the receiver, and the baseline may be flexible. The *DeltaX*, *DeltaY* and *DeltaZ* arguments are ignored in `auto` mode.

In `manual` mode, the user can specify the exact position of the base antenna with respect to the rover antenna in the vehicle reference frame. That reference frame has its X axis pointing to the front of the vehicle, the Y axis pointing to the right, and the Z axis pointing down. Selecting `manual` mode implies that the baseline is rigid. The *DeltaX*, *DeltaY* and *DeltaZ* coordinates are ARP-to-ARP.

Example

In the case of moving-base attitude determination, if the moving-base antenna is located one meter to the left of the rover antenna, and 10 cm below, you should use:

```
COM1> sal, Base, manual, 0, -1, 0.1 <CR>
$R: sal, Base, manual, 0, 1, 0.1
      AntennaLocation, Base, manual, 0.0000, -1.0000, 0.1000
COM1>
```

sao gao	setAntennaOffset getAntennaOffset	Antenna Antenna	DeltaE	DeltaN	DeltaU	Type (20)	SerialNr (20)	SetupID		
		+ Main all	-1000.0000 ... 0.0000 ... 1000.0000 m	-1000.0000 ... 0.0000 ... 1000.0000 m	-1000.0000 ... 0.0000 ... 1000.0000 m	Unknown	Unknown	0 ... 255		

[RxControl: Navigation > Receiver Setup > Antennas](#)

[Web Interface: Configuration > Navigation > Receiver Setup > Antennas](#)

Use these commands to define/inquire the parameters that are associated with the antenna connected to your receiver.

The arguments *DeltaE*, *DeltaN* and *DeltaU* are the offsets of the antenna reference point (ARP) with respect to the marker, in the East, North and Up (ENU) directions respectively, expressed in meters. All absolute positions reported by the receiver are marker positions, obtained by subtracting this offset from the ARP. The purpose is to take into account the fact that the antenna may not be located directly on the surveying point of interest.

Use the argument *Type* to specify the type of your antenna. For best positional accuracy, it is recommended to select a type from the list returned by the command **1stAntennaInfo, Overview**. This is the list of antennas for which the receiver can compensate for phase center variation (see Firmware User Manual for details). If *Type* does not match any entry in the list returned by **1stAntennaInfo, Overview**, the receiver will assume that the phase center variation is zero at all elevations and frequency bands, and the position will not be as accurate. If the antenna name contains whitespaces, it has to be enclosed between double quotes. For proper name matching, it is important to keep the exact same number of whitespaces and the same case as the name returned by **1stAntennaInfo, Overview**.

The argument *SerialNr* is the serial number of your particular antenna. It may contain letters as well as digits (do not forget to enclose the string between double quotes if it contains whitespaces).

The argument *SetupID* is the antenna setup ID as defined in the RTCM standard. It is a parameter for use by the service provider to indicate the particular reference station-antenna combination. The number should be increased whenever a change occurs at the station that affects the antenna phase center variations. Setting *SetupID* to zero means that the values of a standard model type calibration should be used. The value entered for this argument is used to set the setup ID field in the message type 23 of RTCM2.3, and in message types 1007, 1008 and 1033 of RTCM3. It has otherwise no effect on the receiver operation.

Example

```
COM1>  sao, Main, 0.1, 0.0, 1.3, "AERAT2775_159 SPKE", 5684, 0<CR>
$R: sao, Main, 0.1, 0.0, 1.3, "AERAT2775_159  SPKE", 5684, 0
      AntennaOffset, Main, 0.1000, 0.0000, 1.3000, "AERAT2775_159
      SPKE", 5684, 0
COM1>
```


sto	setAttitudeOffset	Heading	Pitch						
gto	getAttitudeOffset								
		0.000 ... 360.000 deg	-90.000 ... 0.000 ... 90.000 deg						

[RxControl: Navigation > Positioning Mode > GNSS Attitude](#)

[Web Interface: Configuration > Navigation > Positioning Mode > GNSS Attitude](#)

Use this command to specify the offsets that the receiver applies to the computed attitude angles.

The attitude of a vehicle (more precisely the heading and pitch angles) can be determined from the orientation of the baseline between two antennas attached to the vehicle. By default, the receiver determines the attitude angles assuming that the baseline between the antenna ARPs is parallel to the longitudinal axis of the vehicle. Attitude biases appear when this is not the case. The user can use this command to provide the value of the biases, such that the receiver can compensate for them before outputting the attitude. The receiver subtracts the offsets specified by the *Heading* and *Pitch* arguments from the attitude angles before encoding them in NMEA or SBF.

Another way to avoid attitude biases is to provide the accurate position of the antennas in the vehicle reference frame. See the command **setAntennaLocation** for more details.

Example

```
COM1> sto, 10.2, -2.5 <CR>
$R: sto, 10.2, -2.5
    AttitudeOffset, 10.200, -2.500
COM1>
```

sdca gdca	setDiffCorrMaxAge getDiffCorrMaxAge	DGPSCorr	RTKCorr	PPPCorr	Iono					
		0.0 ... 400.0 ... 3600.0 sec	0.0 ... 20.0 ... 3600.0 sec	0.0 ... 360.0 ... 3600.0 sec	0.0 ... 600.0 ... 3600.0 sec					

[RxControl: Navigation > Positioning Mode > PPP and Differential Corrections](#)

[Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections](#)

Use these commands to define/inquire the maximum age acceptable for a given differential correction type. A correction is applied only if its age (aka latency) is under the timeout specified with this command and if it is also under the timeout specified with the *MaxAge* argument of the **setDiffCorrUsage** command. In other words, the command **setDiffCorrUsage** sets a global maximum timeout value, while the command **setDiffCorrMaxAge** can force shorter timeout values for certain correction types.

The argument *DGPSCorr* defines the timeout of the range corrections when the PVT is computed in DGPS mode.

The argument *RTKCorr* defines the timeout of the base station code and carrier phase measurements when the PVT is computed in RTK mode.

The argument *PPPCorr* defines the timeout of the wide-area satellite clock and orbit corrections used in PPP mode (only applicable if your receiver supports PPP positioning mode).

The argument *Iono* defines the timeout of the ionospheric corrections (such as transmitted in RTCM2.x MT15) used in DGPS PVT mode.

If the timeout is set to 0, the receiver will never apply the corresponding correction.

Note that this command does not apply to the corrections transmitted by SBAS satellites. For these corrections, the receiver always applies the timeout values prescribed in the DO229 standard.

Example

```
COM1> sdca, 10 <CR>
$R: sdca, 10
    DiffCorrMaxAge, 10.0, 20.0, 300.0, 300.0
COM1>
```

sdcu gdcu	setDiffCorrUsage getDiffCorrUsage	Mode	MaxAge	BaseSelection	BaseID	MovingBase	MaxBase	MaxBaseline		
		LowLatency	0.1 ... 3600.0 sec	auto manual	0 ... 4095	off on	2 ... 5 ... 10	0 ... 2500000 m		

[RxControl: Navigation > Positioning Mode > PPP and Differential Corrections](#)

[Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections](#)

Use these commands to define/inquire the usage of incoming differential corrections in DGPS or RTK rover mode.

The *Mode* argument defines the type of differential solution that will be computed by the receiver. If *LowLatency* is selected, the PVT is computed at the moment local measurements of the receiver are available. At that time, measurements from the base receiver for the same epoch are not yet available due to latency in the transmission, and the PVT is computed with measurements from two different epochs.

The *MaxAge* argument defines the maximum age of the differential corrections to be considered valid. *MaxAge* applies to all types of corrections (DGPS, RTK, satellite orbit, etc), except for those received from a SBAS satellite. See also the command **setDiffCorrMaxAge** to set different maximum ages for different correction types.

The *BaseSelection* argument defines how the receiver should select the base station(s) to be used. If *auto* is selected and the receiver is in DGPS-rover mode, it will use all available base stations. If *auto* is selected and the receiver is in RTK-rover mode, it will automatically select the nearest base station. If *manual* is selected, the receiver will only use the corrections from the base station defined by the *BaseID* argument (in both DGPS and RTK modes).

The *MovingBase* argument defines whether the base station is static or moving.

MaxBase sets the maximum number of base stations to include in the PVT solution in multi-base DGNSS mode.

MaxBaseline sets the maximum baseline length: base stations located beyond the maximum baseline length are excluded from the PVT.

Examples

```
COM1> sdcu, , 5 <CR>
$R: sdcu, , 5
    DiffCorrUsage, LowLatency, 5.0, auto, 0, off, 20, 20000000
COM1>

COM1> gdcu <CR>
$R: gdcu
    DiffCorrUsage, LowLatency, 5.0, auto, 0, off, 20, 20000000
COM1>
```

sem gem	setElevationMask getElevationMask	Engine Engine	Mask							
		+ Tracking + PVT all	-90 ... 0 ... 90 deg							

RxControl: Navigation > Receiver Operation > Masks

Web Interface: Configuration > Navigation > Receiver Operation > Masks

Use these commands to set or get the elevation mask in degrees. There are two masks defined: a tracking mask and a PVT mask.

Satellites under the tracking elevation mask are not tracked, and therefore there is no measurement, nor navigation data available from them. The tracking elevation mask does not apply to SBAS satellites: SBAS satellites are generally used to supply corrections and it is undesirable to make the availability of SBAS corrections dependent on the satellite elevation. The tracking elevation mask does not apply to satellites that are manually assigned with the **setChannelAllocation** command.

Satellite under the PVT mask are not included in the PVT solution, though they still provide measurements and their navigation data is still decoded and used. The PVT elevation mask do apply to the SBAS satellites: the ranges to SBAS satellites under the elevation mask are not used in the PVT, but the SBAS corrections are still decoded and potentially used in the PVT.

Although possible, it does not make sense to select a higher elevation mask for the tracking than for the PVT, as, obviously, a satellite which is not tracked cannot be included in the PVT.

The mask can be negative to allow the receiver to track satellites below the horizon. This can happen in case the receiver is located at high altitudes or if the signal is refracted through the atmosphere.

Examples

```
COM1> sem, PVT, 10 <CR>
$R: sem, PVT, 10
      ElevationMask, PVT, 10
COM1>

COM1> gem <CR>
$R: gem
      ElevationMask, Tracking, 0
      ElevationMask, PVT, 10
COM1>
```

sfr	setFixReliability	Engine	SearchVolume	Ratio						
gfr	getFixReliability	Engine								
		+ RTK	0.001 ... 0.200	1.00 ... 4.40						
		all	... 10.000	... 20.00						

[RxControl: Navigation > Receiver Operation > Position > Ambiguities](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Ambiguities](#)

Use these commands to define/inquire the criteria that control the ambiguity fixing process of the RTK and/or attitude-determination engines.

The ambiguity fixing algorithm searches for the most likely integer carrier phase ambiguity set within the prescribed *SearchVolume*.

All integer ambiguity vectors contained in the search volume are sorted according to their distance to the float ambiguities. The candidate with the lowest value will be selected as the most likely ambiguity set. The likelihood of this candidate with respect to other candidates is characterized by the ratio between the best and the second-best candidate. If this ratio is lower than the prescribed threshold (*Ratio*, the third argument), the candidate is rejected and the ambiguity search will restart at the next epoch.

Lowering *SearchVolume* and increasing *Ratio* will increase the time to fix but also decrease the probability of fixing incorrect ambiguities.

This command should only be used by expert users who understand the consequences of modifying the default values. It is strongly recommended to leave all settings to their default values.

Examples

```
COM1> sfr, RTK, 0.2 <CR>
$R: sfr, RTK, 0.2
    FixReliability, RTK, 0.200, 4.40
COM1>
```

```
COM1> gfr, RTK <CR>
$R: gfr, RTK
    FixReliability, RTK, 0.200, 4.40
COM1>
```

sgd ggd	setGeodeticDatum getGeodeticDatum	TargetDatum								
		WGS84 ETRS89 NAD83 NAD83_PA NAD83_MA GDA94 Default User1 User2								

RxControl: [Navigation > Receiver Operation > Position > Datum](#)

Web Interface: [Configuration > Navigation > Receiver Operation > Position > Datum](#)

Use this command to define the datum to which you want the coordinates to refer.

TargetDatum	Description
WGS84	Equivalent to Default
ETRS89	European ETRS89 (ETRF2000 realization)
NAD83	NAD83(2011), North American Datum (2011)
NAD83_PA	NAD83(PA11), North American Datum, Pacific plate (2011)
NAD83_MA	NAD83(MA11), North American Datum, Marianas plate (2011)
GDA94	GDA94(2010), Geocentric Datum of Australia (2010)
Default	Default datum, which depends on the positioning mode as explained below
User1	First user-defined datum. The corresponding transformation parameters must be specified by the setUserDatum and setUserDatumVel commands, while the corresponding ellipsoid must be defined by the setUserEllipsoid command.
User2	Second user-defined datum

By default (argument *TargetDatum* set to `Default`), the datum depends on the positioning mode. For standalone and SBAS positioning, the coordinates refer to a global datum: WGS84 or ITRF (recent realisations of WGS84 and ITRF are closely aligned and the receiver considers them equivalent). When using corrections from the LBAS1 service (multi-base DGNSS or PPP), the coordinates refer to ITRF. When using DGNSS or RTK corrections from a local DGNSS/RTK provider, the datum is defined by the coordinates of the base station.

With this command, the user can select the datum the coordinates should refer to. In case you are using corrections from a local DGNSS/RTK provider, the datum to be specified here must be the datum used by your correction provider.

When a non-default datum is selected, the receiver transforms all the WGS84/ITRF coordinates to the specified datum. Positions obtained using local or regional DGNSS/RTK corrections are not transformed, as it is assumed that the selected datum is the one used by the DGNSS/RTK provider.

In the current firmware version, the `WGS84` value for the *TargetDatum* argument has no effect, but it is kept for backwards compatibility reasons. Setting *TargetDatum* to `WGS84` is equivalent to setting it to `Default`.

Example

```
COM1> sgd, ETRS89 <CR>  
$R: sgd, ETRS89  
      GeodeticDatum, ETRS89  
COM1>
```


sgu	setGeoidUndulation	Mode	Undulation							
ggu	getGeoidUndulation									
		auto	-250.0 ... 0.0							
		manual	... 250.0 m							

[RxControl: Navigation > Receiver Operation > Position > Earth Models](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Earth Models](#)

Use these commands to define/inquire the geoid undulation at the receiver position. The geoid undulation specifies the local difference between the geoid and the WGS84 ellipsoid.

If *Mode* is set to `auto`, the receiver computes the geoid undulation with respect to the WGS84 ellipsoid using the model defined in 'Technical Characteristics of the NAVSTAR GPS, NATO, June 1991', regardless of the datum specified with the `setGeodeticDatum` command. In `auto` mode, the *Undulation* argument is ignored.

The geoid undulation is included in the `PVTCartesian` and the `PVTGeodetic` SBF blocks and in the NMEA position messages.

Examples

```
COM1> sgu, manual, 25.3 <CR>
$R: sgu, manual, 25.3
    GeoidUndulation, manual, 25.3
COM1>
```

```
COM1> ggu <CR>
$R: ggu
    GeoidUndulation, manual, 25.3
COM1>
```

sga	setGNSSAttitude	Source								
gga	getGNSSAttitude	none								
		MovingBase								

RxControl: Navigation > Positioning Mode > GNSS Attitude

Web Interface: Configuration > Navigation > Positioning Mode > GNSS Attitude

Use this command to define/inquire the way GNSS-based attitude is computed.

The attitude of a vehicle (more precisely the heading and pitch angles) can be determined from the orientation of the baseline between two antennas attached to the vehicle. See also the **setAntennaLocation** command.

The *Source* argument specifies how to compute the GNSS-based attitude:

Source	Description
none	GNSS attitude computation is disabled.
MovingBase	Attitude is computed from the baseline between antennas connected to two receivers configured in moving-base RTK operation (moving-base attitude).

Example

```
COM1> sga, MovingBase <CR>
$R: sga, MovingBase
      GNSSAttitude, MovingBase, Fixed
COM1>
```

shm	setHealthMask	Engine	Mask							
ghm	getHealthMask	Engine	Engine							
		+ Tracking	off							
		+ PVT	on							
		all								

[RxControl: Navigation > Receiver Operation > Masks](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Masks](#)

Use these commands to define/inquire whether measurements should be produced for unhealthy satellite signals, and whether these measurements should be included in the PVT solution. All satellite signals of which the health is unknown to the receiver (because the health information has not been decoded yet or is not transmitted) are considered healthy.

If *Mask* is *on* for the *Tracking* engine, no measurements are generated for unhealthy signals, but these signals will remain internally tracked and their navigation data will be decoded and processed. This is to ensure immediate reaction in the event that the signal would become healthy again.

If *Mask* is *on* for the *PVT* engine, measurements from unhealthy signals are not included in the PVT. Setting this mask to *off* must be done with caution: including a non-healthy signal in the PVT computation may lead to unpredictable behaviour of the receiver.

Although possible, it does not make sense to enable unhealthy satellites for the PVT if they are disabled for tracking.

Examples

To track unhealthy satellites/signals, use:

```
COM1> shm, Tracking, off <CR>
$R: shm, Tracking, off
    HealthMask, Tracking, off
COM1>
```

```
COM1> ghm <CR>
$R: ghm
    HealthMask, Tracking, off
    HealthMask, PVT, on
COM1>
```

sim	setIonosphereModel	Model								
gim	getIonosphereModel	auto								
		off								
		Klobuchar								
		SBAS								
		MultiFreq								

RxControl: Navigation > Receiver Operation > Position > Atmosphere
Web Interface: Configuration > Navigation > Receiver Operation > Position > Atmosphere

Use these commands to define/inquire the type of model used to correct ionospheric errors in the PVT computation. The following models are available:

Model	Description
auto	With this selection, the receiver will, based on the available information, automatically select the best model on a satellite to satellite basis.
off	The receiver will not correct measurements for the ionospheric delay. This may be desirable if the receiver is connected to a GNSS signal simulator.
Klobuchar	This model uses the parameters as transmitted by the GPS satellites to compute the ionospheric delays.
SBAS	This model complies with the DO229 standard [1]: it uses the near real-time ionospheric delays transmitted by the SBAS satellites in MT18 and MT26. If no such message has been received, the Klobuchar model is selected automatically.
MultiFreq	This model uses a combination of measurements on different carriers to accurately estimate ionospheric delays. It requires the availability of at least dual-frequency measurements.

Unless the model is set to `auto`, the receiver uses the same model for all satellites, e.g. if the `Klobuchar` model is requested, the Klobuchar parameters transmitted by GPS satellites are used for all tracked satellites, regardless of their constellation.

If not enough data is available to apply the prescribed model to a given satellite (for instance if only single-frequency measurements are available and the model is set to `MultiFreq`), the satellite in question will be discarded from the PVT. Under most circumstances, it is recommended to leave the model to `auto`.

Examples

To disable the compensation for ionospheric delays, use:

```
COM1> sim, off <CR>
$R: sim, off
      IonosphereModel, off
COM1>
```

```
COM1> gim <CR>
$R: gim
      IonosphereModel, off
COM1>
```

smv	setMagneticVariance	Mode	Variance							
gmV	getMagneticVariance									
		auto	-180.0 ... 0.0							
		manual	... 180.0 deg							

[RxControl: Navigation > Receiver Operation > Position > Earth Models](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Earth Models](#)

Use these commands to define the magnetic variance (a.k.a. magnetic declination) at the current position. The magnetic variance specifies the local offset of the direction to the magnetic north with respect to the geographic north. The variance is positive when the magnetic north is east of the geographic north.

By default (the argument *Mode* is set to `auto`), the receiver automatically computes the variance according to the International Geomagnetic Reference Field (IGRF) model, using the IGRF2010 coefficients corrected for the secular variation.

Note that the magnetic variance is used solely in the generation of NMEA messages.

Examples

```
COM1> smv, manual, 1.1 <CR>
$R: smv, manual, 1.1
    MagneticVariance, manual, 1.1
COM1>
```

```
COM1> gmV <CR>
$R: gmV
    MagneticVariance, manual, 1.1
COM1>
```

snrc	setNetworkRTKConfig	NetworkType								
gnrc	getNetworkRTKConfig									
		auto								
		VRS								

RxControl: Navigation > Positioning Mode > PPP and Differential Corrections

Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections

Use these commands to define/inquire the type of the RTK network providing the differential corrections.

In most cases, it is recommended to leave the *Type* argument to `auto` to let the receiver autodetect the network type. For some types of VRS networks (especially for those having long baselines between the base stations), optimal performance is obtained by forcing the type to `VRS`.

Example

```
COM1> snrc, VRS <CR>
$R: snrc, VRS
NetworkRTKConfig, VRS
COM1>
```

snl gnl	setNWALevels getNWALevels	Mode	HAL	VAL						
		off on	0.00 ... 1.20 ... 1000.00 m	0.00 ... 2.00 ... 1000.00 m						

[RxControl: Navigation > Receiver Operation > Position > Integrity](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Integrity](#)

Use this command to set the navigation warning algorithm alert levels.

When *Mode* is "on", the receiver compares the 2DRMS horizontal and vertical accuracies of the position to the specified *HAL* and *VAL* values respectively. If one of the 2DRMS accuracies exceeds the corresponding alert limit, an "accuracy-not-met" flag is set in the *AlertFlag* of the *PVTCartesian* and *PVTGeodetic* SBF blocks.

When *Mode* is *off* (default), the accuracy test is disabled.

Example

```
COM1> snl, on, 20, 30<CR>
$R: snl, on, 20, 30
    NWALevels, on, 20.0, 30.0
COM1>
```


epss gps	exePPPSetSeedGeod getPPPSetSeedGeod	Latitude	Longitude	Altitude	Datum					
		-90.000000000 ... 0.000000000 ... 90.000000000 deg	-180.000000000 ... 0.000000000 ... 180.000000000 deg	-1000.0000 ... 0.0000 ... 30000.0000 m	WGS84 ETRS89 NAD83 NAD83_PA NAD83_MA GDA94 User1 User2 Other					

[RxControl: Navigation > Receiver Initialization > PPP Set Seed](#)

[Web Interface: Configuration > Navigation > Receiver Initialization > PPP Set Seed](#)

Use this command to seed the PPP engine with the specified position. The geodetic coordinates in the *Latitude*, *Longitude* and *Altitude* arguments are that of the marker, and not of the ARP. The argument *Longitude* is positive to the east of Greenwich.

The datum to which the coordinates refer must be specified with the *Datum* argument:

Datum	Description
WGS84	WGS84 or ITRFxx (the receiver does not make a distinction between them)
ETRS89	European ETRS89 (ETRF2000 realization)
NAD83	NAD83(2011), North American Datum (2011)
NAD83_PA	NAD83(PA11), North American Datum, Pacific plate (2011)
NAD83_MA	NAD83(MA11), North American Datum, Marianas plate (2011)
GDA94	GDA94(2010), Geocentric Datum of Australia (2010)
User1	First user-defined datum. The corresponding transformation parameters must be specified by the setUserDatum and setUserDatumVel commands, while the corresponding ellipsoid must be defined by the setUserEllipsoid command.
User2	Second user-defined datum
Other	Datum not in the list or unknown

The receiver applies the necessary transformations to ITRF (the datum used by the PPP engine) automatically. If the *Datum* argument is set to *Other*, no datum transformation is applied.

It is the user's responsibility to ensure that the specified marker position is centimeter-accurate and that the datum is properly specified. Inaccurate seeding will result in a long convergence time, and/or in an incorrect estimate of the variance/covariance matrix of the PPP solution.

At the moment when the command is entered, the receiver must be static. To prevent gross errors, the command is ignored when the seed significantly differs from the current position computed by the receiver, or when the current velocity is not zero.

In the event that the command is issued when the receiver is already in PPP mode, the PPP filter is reset and re-seeded.

Example

```
COM1> epss, 4.5, 3.568, 0.1 <CR>  
$R: epss, 4.5, 3.568, 0.1  
    PPPSetSeedGeod, 4.500000000, 3.568000000, 0.1000  
COM1>
```

spm gpm	setPVTMode getPVTMode	Mode	RoverMode	StaticPosition	ExtSensorIntegrati					
		Static	+ StandAlone	auto	off					
		Rover	+ SBAS	Geodetic1	SIGIL					
			+ DGPS	Geodetic2						
			+ RTKFloat	Geodetic3						
			+ RTKFixed	Geodetic4						
			+ PPP	Geodetic5						
			+ RTK	Cartesian1						
			all	Cartesian2						
				Cartesian3						
				Cartesian4						
				Cartesian5						

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration > Navigation > Positioning Mode > PVT Mode

Use these commands to define/inquire the PVT mode of the receiver. The argument *Mode* specifies the general positioning mode. If *Rover* is selected, the receiver will assume that the receiver is moving and compute the best PVT allowed by the *RoverMode* argument. If *Static* is selected, the receiver will assume that it is fixed and will use the position defined by the *StaticPosition* argument.

The argument *RoverMode* specifies the allowed PVT modes when the receiver is operating in *Rover* mode. Different modes can be combined with the "+" operator. Refer to the "Operation Details" chapter of the Firmware User Manual for a description of the PVT modes. The value *RTK* is an alias for *RTKFloat+RTKFixed*. When more than one mode is enabled in *RoverMode*, the receiver automatically selects the mode that provides the most accurate solution with the available data.

The position provided in the *StaticPosition* argument must be defined with the **setStaticPosCartesian** or the **setStaticPosGeodetic** commands. If the value *auto* is selected for this argument, the receiver will wait till a reliable PVT solution is available and it will use that solution as static position. In *auto* mode, the static coordinates computed by the receiver refer to the datum specified with the **setGeodeticDatum** argument.

The argument *ExtSensorIntegration* defines how to compute an INS/GNSS integrated navigation and attitude solution in case an IMU sensor is connected to the receiver. The following options are available:

ExtSensorIntegration	Description
off	INS data not used by the PVT (GNSS-only solution)
SIGIL	Proprietary

Examples

```
COM1> spm, Rover, StandAlone+RTK <CR>
$R: spm, Rover, StandAlone+RTK
    PVTMode, Rover, StandAlone+RTK, auto, off
COM1>
```

To set up a fixed base station at a known location, use the following:

```
COM1> sspg, Geodetic1, 50.5209, 4.4245, 113.3 <CR>
$R: sspg, Geodetic1, 50.5209, 4.4245, 113.3
    StaticPosGeodetic, Geodetic1, 50.52090000, 4.42450000, 113.3000
COM1> spm, Static, , Geodetic1 <CR>
```

```
$R: spm, Static, , Geodetic1  
    PVTMode, Static, StandAlone+RTK, Geodetic1, off  
COM1>
```

srl gri	setRAIMLevels getRAIMLevels	Mode	Pfa	Pmd	Reliability					
		off on	-12 ... -4 ... -1	-12 ... -4 ... -1	-12 ... -3 ... -1					

[RxControl: Navigation > Receiver Operation > Position > Integrity](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Integrity](#)

Use these commands to define/inquire the parameters of the Receiver Autonomous Integrity Monitoring (RAIM) algorithm in rover PVT mode. Refer to the Firmware User Manual for a description of RAIM in your receiver.

The *Mode* argument acts as an on/off switch: it determines whether RAIM is active or not. If *Mode* is set to `off`, the test statistics are still computed and the results are available in the `RAIMStatistics` SBF block. However, the outcome of the tests has no effect on the positional output: there is no attempt to remove outliers from the PVT.

The *Pfa* argument sets the probability of false alarm of the w-test used in the "identification" step of the RAIM algorithm. Increasing this parameter increases the integrity but may reduce the availability of the positional solution.

The *Pmd* argument sets the probability of missed detection, which the receiver uses to compute the Minimal Detectable Bias and hence the XERL values.

The *Reliability* argument sets the probability of false alarm of the Overall Model test used in the "detection" step of the RAIM algorithm.

The value to be provided in the *Pfa*, *Pmd* and *Reliability* arguments are the base-10 logarithms of the desired probabilities. For instance, if you want a probability of false alarm of 1e-6, you have to set the *Pfa* argument to -6.

Note that this command has no effect when the receiver is configured in static PVT mode with the `setPVTMode, static` command.

Examples

To configure the receiver outlier detection with a probability of 1e-4 (0.01%) that a false alarm will be raised (type I error), a probability of 1e-4 (0.01%) that an outlier will be missed (type II error) and an Overall Model probability of false alarm of 1e-6 (0.0001%), use:

```
COM1> srl, on, -4, -4, -6 <CR>
$R: srl, on, -4, -4, -6
    RAIMLevels, on, -4, -4, -6
COM1>
```

To disable the outlier detection, use:

```
COM1> srl, off <CR>
$R: srl, off
    RAIMLevels, off, -4, -4, -6
COM1>
```

srd grd	setReceiverDynamics getReceiverDynamics	Level	Motion							
		Max	Static							
		High	Quasistatic							
		Moderate	Pedestrian							
		Low	Automotive							
			RaceCar							
			HeavyMachinery							
			UAV							
			Unlimited							

RxControl: Navigation > Receiver Operation > Position > Motion

Web Interface: Configuration > Navigation > Receiver Operation > Position > Motion

Use these commands to set/inquire the type of the dynamics the GNSS antenna is subjected to. The receiver adapts internal parameters for optimal performance for the selected type of dynamics.

The *Level* argument indicates the level of short-term acceleration and jerk the antenna is subjected to. That argument has effect on the navigation Kalman filter and on the tracking loops (only if the loops are in adaptive mode, see the **setTrackingLoopParameters** command). Setting this argument to `low` will reduce the noise on the GNSS measurements and PVT at the expense of filtering out rapid dynamic changes. Setting it to `high` will allow more dynamics to be visible at the expense of noise. The `max` level is primarily meant for test purposes: in that level, the Kalman filter is disabled and the receiver computes epoch-by-epoch independent PVT solutions.

Note that rapid displacements such as the ones caused by shocks, drops, oscillations or vibrations lead to high jerk values, even if the amplitude of the motion is not larger than a few centimeters. It is recommended to set *Level* to `high` for antennas subjected to that type of displacements.

The *Motion* argument defines the type of motion the antenna is subjected to.

Motion	Description
Static	Fixed base and reference stations.
Quasistatic	Low speed, limited area motion typical of surveying applications.
Pedestrian	Low speed (<7m/s) motion. E.g. pedestrians, low-speed land vehicles, ...
Automotive	Medium speed (<50m/s) motion. E.g. passenger cars, rail vehicles, ...
RaceCar	High speed terrestrial vehicle. E.g. race cars, ...
HeavyMachinery	Construction equipment, tractors, ...
UAV	Unmanned Aerial Vehicle.
Unlimited	Unconstrained motion.

Example

```
COM1> srd, Max <CR>
$R: srd, Max
ReceiverDynamics, Max, Automotive
COM1>
```

ernf grnf	exeResetNavFilter getResetNavFilter	Level								
		+ PVT + AmbRTK all								

RxControl: Navigation > Receiver Initialization > Reset Navigation Filter

Web Interface: Configuration > Navigation > Receiver Initialization > Reset Navigation Filter

Use this command to reset the different navigation filters in the receiver. The user can reset each navigation filter independently or together with the value `all`.

The following values for *Level* are defined:

Level	Description
PVT	Reset the whole PVT filter such that all previous positioning information is discarded, including the RTK ambiguities and the INS/GNSS integration filter when applicable.
AmbRTK	Only reset the ambiguities used in RTK positioning to float status.

Example

```
COM1> ernf, PVT <CR>
$R: ernf, PVT
      ResetNavFilter, PVT
COM1>
```


ssu gsu	setSatelliteUsage getSatelliteUsage	Satellite								
		none + G01 ... G32 + R01 ... R24 + R25 + R26 + R27 + R28 + R29 + R30 + E01 ... E32 + S120 ... S158 + C01 ... C37 + GPS + GLONASS + GALILEO + SBAS + COMPASS all								

RxControl: Navigation > Advanced User Settings > PVT > Satellite Usage

Web Interface: Configuration > Navigation > Advanced User Settings > PVT > Satellite Usage

Use these commands to define/inquire which satellites are allowed to be included in the PVT computation. It is possible to enable or disable a single satellite (e.g. G01 for GPS PRN1), or a whole constellation. Gxx, Exx, Rxx, Cxx and Sxxx refer to a GPS, Galileo, GLONASS, Compass/BeiDou and SBAS satellite respectively. GLONASS satellites must be referenced by their slot number in this command.

Examples

To only use GPS measurements in the PVT computation, use:

```
COM1> ssu, GPS <CR>
$R: ssu, GPS
SatelliteUsage, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26
+G27+G28+G29+G30+G31+G32
COM1>
```

To add the usage of SBAS measurements in the PVT, use:

```
COM1> ssu, +SBAS <CR>
$R: ssu, +SBAS
SatelliteUsage, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26
+G27+G28+G29+G30+G31+G32+S120+S121+S122+S123+S124+S125+S126
...
COM1>
```

To remove the measurement of one satellite from the PVT, use:

```
COM1> ssu, -S120 <CR>
$R: ssu, -S120
SatelliteUsage, G01+G02+G03+G04+G05+G06+G07+G08+G09+G10+G11
+G12+G13+G14+G15+G16+G17+G18+G19+G20+G21+G22+G23+G24+G25+G26
+G27+G28+G29+G30+G31+G32+S121+S122+S123+S124+S125+S126+S127
...
```

COM1>

ssbc gsbc	setSBASCORRECTIONS getSBASCORRECTIONS	Satellite	SISMode	NavMode	DO229Version					
		auto EGNOS WAAS MSAS S120 ... S158	Test <u>Operational</u>	EnRoute PrecApp <u>MixedSystems</u>	auto DO229C					

RxControl: Navigation > Positioning Mode > SBAS Corrections

Web Interface: Configuration > Navigation > Positioning Mode > SBAS Corrections

Use these commands to define/inquire the details on the usage of SBAS data in the PVT computation. This command does not define whether SBAS corrections are to be used or not in the PVT (this is done by the **setPVTMode** command), but it specifies how these corrections should be used.

The *Satellite* argument defines the provider of SBAS corrections, being either an individual satellite or a satellite system. If **EGNOS**, **WAAS** or **MSAS** is selected, the receiver restricts the automatic selection of a satellite to those that are part of the EGNOS, WAAS or MSAS system. When **auto** is selected for the *Satellite* argument, the receiver will automatically select a satellite on the basis of the location of the receiver and on the availability of SBAS corrections.

The *SISMode* argument defines the interpretation of a "Do Not Use for Safety Applications" message. When set to **Operational**, the receiver will discard all SBAS corrections received from a satellite upon reception of a MT00 from that satellite. Note that MT02 content encoded in a MT00 message will be interpreted by the receiver as a MT02 message: only MT00 with all '0' symbols will be interpreted as a true "Do Not Use for Safety Applications". When the argument *SISMode* is set to **Test**, the receiver will ignore the reception of a "Do Not Use for Safety Applications" message. This provides the possibility to use a signal from a SBAS system in test mode.

The DO 229 standard, which has its origin in aviation, makes a distinction between two positioning applications: en-route and precision approach. The choice between both applications influences the length of the interval during which the SBAS corrections are valid. During a precision approach the validity of the data is much shorter. The receiver can operate in both modes, which is controlled by the *NavMode* argument.

In **EnRoute** or in **PrecApp** mode, the receiver only uses the satellite systems for which SBAS corrections are available. For best positioning accuracy, it is typically preferable to include all satellites in the position computation, even if they are not corrected by SBAS. This is achieved by the **MixedSystems** mode.

The *DO229Version* argument can be used to specify which version of the DO 229 standard to conform to.

Example

To force the receiver to use corrections from PRN 122 and ignore message MT00:

```
COM1> ssbc, S122, Test <CR>
$R: ssbc, S122, Test
      SBASCORRECTIONS, S122, Test, EnRoute, auto
COM1>
```

snu gnu	setSignalUsage getSignalUsage	PVT	NavData							
		+GPSL1CA +GPSL1PY +GPSL2PY +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +GEOL1 +GEOL5 +CMPL1 +CMPE5b all	+GPSL1CA +GPSL1PY +GPSL2PY +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +GEOL1 +GEOL5 +CMPL1 +CMPE5b +QZSL1CA +QZSL2C +QZSL5 all							

RxControl: [Navigation > Advanced User Settings > PVT > Signal Usage](#)

Web Interface: [Configuration > Navigation > Advanced User Settings > PVT > Signal Usage](#)

Use these commands to define/inquire which signal types are used by the receiver.

The *PVT* argument lists the signals that can be used by the PVT. Removing a signal from the list will disable the usage of the corresponding range, phase & Doppler measurements in the PVT computation.

The *NavData* argument lists the signals for which the receiver is allowed to decode the navigation message. Removing a signal from the list will disable the usage of the corresponding navigation information (ephemeris, ionosphere parameters ...).

Example

To force the receiver to only use the L1 GPS C/A measurements and navigation information in the PVT solution, use:

```
COM1> snu, GPSL1CA, GPSL1CA <CR>
$R: snu, GPSL1CA, GPSL1CA
SignalUsage, GPSL1CA, GPSL1CA
COM1>
```

sspc gspc	setStaticPosCartesian getStaticPosCartesian	Position Position	X	Y	Z	Datum				
		+ Cartesian1	-8000000.0000	-8000000.0000	-8000000.0000	WGS84				
		+ Cartesian2	... 0.0000	... 0.0000	... 0.0000	ETRS89				
		+ Cartesian3	... 8000000.0000	... 8000000.0000	... 8000000.0000	NAD83				
		+ Cartesian4	m	m	m	NAD83_PA				
		+ Cartesian5				NAD83_MA				
		all				GDA94				
						User1				
						User2				
						Other				

RxControl: Navigation > Positioning Mode > PVT Mode

Web Interface: Configuration > Navigation > Positioning Mode > PVT Mode

Use these commands to define/inquire a set of Cartesian coordinates. This command should be used in conjunction with the **setPVTMode** command to specify a base station position. The Cartesian coordinates in the X, Y and Z arguments must refer to the antenna reference point (ARP), and not to the marker.

The argument *Datum* specifies the datum to which the coordinates refer:

Datum	Description
WGS84	WGS84 or ITRFxx (the receiver does not make a distinction between them)
ETRS89	European ETRS89 (ETRF2000 realization)
NAD83	NAD83(2011), North American Datum (2011)
NAD83_PA	NAD83(PA11), North American Datum, Pacific plate (2011)
NAD83_MA	NAD83(MA11), North American Datum, Marianas plate (2011)
GDA94	GDA94(2010), Geocentric Datum of Australia (2010)
User1	First user-defined datum. The corresponding transformation parameters must be specified by the setUserDatum and setUserDatumVel commands, while the corresponding ellipsoid must be defined by the setUserEllipsoid command.
User2	Second user-defined datum
Other	Datum not in the list or unknown

Example

To set up a static base station in Cartesian coordinates:

```
COM1> sspc, Cartesian1, 4019952.028, 331452.954, 4924307.458 <CR>
$R: sspc, Cartesian1, 4019952.028, 331452.954, 4924307.458
    StaticPosCartesian, Cartesian1, 4019952.0280, 331452.9540,
    4924307.4580, WGS84
COM1> spm, Static, , Cartesian1 <CR>
$R: spm, Static, , Cartesian1
    PVTMode, Static, StandAlone+SBAS+DGPS+RTKFloat+RTKFixed,
    Cartesian1
COM1>
```

sspg gspg	setStaticPosGeodetic getStaticPosGeodetic	Position Position	Latitude	Longitude	Altitude	Datum				
		+ Geodetic1 + Geodetic2 + Geodetic3 + Geodetic4 + Geodetic5 all	-90.000000000 ... 0.000000000 ... 90.000000000 deg	-180.000000000 ... 0.000000000 ... 180.000000000 deg	-1000.0000 ... 0.0000 ... 30000.0000 m	WGS84 ETRS89 NAD83 NAD83_PA NAD83_MA GDA94 User1 User2 Other				

[RxControl: Navigation > Positioning Mode > PVT Mode](#)

[Web Interface: Configuration > Navigation > Positioning Mode > PVT Mode](#)

Use these commands to define/inquire a set of geodetic coordinates. This command should be used in conjunction with the **setPVTMode** command to specify a base station position. The geodetic coordinates in the *Latitude*, *Longitude* and *Altitude* arguments must refer to the antenna reference point (ARP), and not to the marker.

The argument *Datum* specifies the datum to which the coordinates refer. See the **setStaticPosCartesian** command for a short description of the supported datums.

Example

To set up a static base station in geodetic coordinates:

```
COM1> sspg, Geodetic1, 50.86696443, 4.71347657, 114.880 <CR>
$R: sspg, Geodetic1, 50.86696443, 4.71347657, 114.880
    StaticPosGeodetic, Geodetic1, 50.86696443, 4.71347657, 114.8800,
    WGS84
COM1> spm, Static, , Geodetic1 <CR>
$R: spm, Static, , Geodetic1
    PVTMode, Static, StandAlone+SBAS+DGPS+RTKFloat+RTKFixed,
    Geodetic1
COM1>
```

sts	setTimingSystem	System								
gts	getTimingSystem									
		GST								
		GPS								

[RxControl: Navigation > Receiver Operation > Timing](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Timing](#)

Use these commands to define/inquire the time system in which the receiver should operate. Galileo System Time (GST) is only supported on Galileo-enabled receivers. The selected *System* time will be used as reference time for clock bias computation.

Note that at least one satellite of the selected system (GPS or Galileo) must be visible and tracked by the receiver. Otherwise no PVT will be computed.

Examples

```
COM1> sts, GPS <CR>
$R: sts, GPS
      TimingSystem, GPS
COM1>
```

```
COM1> gts <CR>
$R: gts
      TimingSystem, GPS
COM1>
```


stm gtm	setTroposphereModel getTroposphereModel	ZenithModel	MappingModel							
		off Saastamoinen MOPS	Niell MOPS							

[RxControl: Navigation > Receiver Operation > Position > Atmosphere](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Atmosphere](#)

Use these commands to define/inquire the type of model used to correct tropospheric errors in the PVT computation.

The *ZenithModel* parameter indicates which model the receiver uses to compute the dry and wet delays for radio signals at 90 degree elevation. The modelled zenith tropospheric delay depends on assumptions for the local air total pressure, the water vapour pressure and the mean temperature. The following zenith models are defined:

ZenithModel	Description
off	The measurements will not be corrected for the troposphere delay. This may be desirable if the receiver is connected to a GNSS signal simulator.
Saastamoinen	Saastamoinen, J. (1973). "Contributions to the theory of atmospheric refraction". In three parts. Bulletin Geodesique, No 105, pp. 279-298; No 106, pp. 383-397; No. 107, pp. 13-34.
MOPS	Minimum Operational Performance Standards for Global Positioning/Wide Area Augmentation System Airborne Equipment RTCA/DO-229C, November 28, 2001.

The *Saastamoinen* model uses user-provided values of air temperature, total air pressure referenced to the Mean Sea Level and relative humidity (see **setTroposphereParameters** command) and estimates actual values adjusted to the receiver height.

The MOPS model neglects the user-provided values and instead assumes a seasonal model for all the climatic parameters. Local tropospheric conditions are estimated based on the coordinates and time of the year.

The use of the *Saastamoinen* model can be recommended if external information on temperature, pressure, humidity is available. Otherwise it is advisable to rely on climate models.

The zenith delay is mapped to the current elevation for each satellite using the requested *MappingModel*. The following mapping models are defined:

MappingModel	Description
Niell	Niell, A.E. (1996). Global Mapping Functions for the atmosphere delay at radio wavelengths, Journal of Geophysical Research, Vol. 101, No. B2, pp. 3227-3246.
MOPS	Minimum Operational Performance Standards for Global Positioning/Wide Area Augmentation System Airborne Equipment RTCA/DO-229C, November 28, 2001.

Examples

```
COM1> stm, MOPS, MOPS <CR>
```

```
$R: stm, MOPS, MOPS  
    TroposphereModel, MOPS, MOPS  
COM1>
```

```
COM1> gtm <CR>  
$R: gtm  
    TroposphereModel, MOPS, MOPS  
COM1>
```

stp gtp	setTroposphereParameters getTroposphereParameters	Temperature	Pressure	Humidity						
		-100.0 ... 15.0 ... 100.0 degC	800.00 ... 1013.25 ... 1500.00 hPa	0 ... 50 ... 100 %						

RxControl: Navigation > Receiver Operation > Position > Atmosphere

Web Interface: Configuration > Navigation > Receiver Operation > Position > Atmosphere

Use these commands to define/inquire the climate parameters to be used when the zenith troposphere is estimated using the Saastamoinen model (see the **setTroposphereModel** command).

The troposphere model assumes the climate parameters to be valid for a receiver located at the Mean Sea Level (MSL). If you want to use your receiver with a weather station, you have to convert the measured *Temperature*, *Pressure* and *Humidity* to MSL.

Example

```
COM1> stp, 25, 1013, 60<CR>
$R: stp, 25, 1013, 60
    TroposphereParameters, 25.0, 1013.00, 60
COM1>
```

sud gud	setUserDatum getUserDatum	Datum Datum	T_x	T_y	T_z	R_x	R_y	R_z	D	
		+ User1	-2000000.00	-2000000.00	-2000000.00	-100.0000	-100.0000	-100.0000	-100.00000	
		+ User2	... 0.00	... 0.00	... 0.00	... 0.0000	... 0.0000	... 0.0000	... 0.00000	
		all	... 2000000.00 mm	... 2000000.00 mm	... 2000000.00 mm	... 100.0000 mas	... 100.0000 mas	... 100.0000 mas	... 100.00000 ppb	

RxControl: Navigation > Receiver Operation > Position > Datum

Web Interface: Configuration > Navigation > Receiver Operation > Position > Datum

Use these commands to define datum transformation parameters from the global WGS84/ITRF datum to the user datum identified by the first argument.

The receiver applies the linearized form of the Helmert similarity transformation. The coordinates in WGS84/ITRF are transformed to the user datum using the following formula:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{User} = \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix} + \begin{pmatrix} D+1 & -R_z & R_y \\ R_z & D+1 & -R_x \\ -R_y & R_x & D+1 \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{WGS84/ITRF}$$

where T_x , T_y and T_z are the three translation components, R_x , R_y and R_z are the rotation angles and D is the scale factor. Note that the rotation angles are expressed in radians in the above formula, but they must be provided in milliarcsecond (1 mas = $2\pi/360/3600000$ radians) in the arguments of the command. The sign convention corresponds to that of the IERS Conventions (2010), Technical Note No. 36.

The time derivative of the transformation parameters can be specified with the command **setUserDatumVel**.

For the receiver to apply the transformation parameters, the corresponding user datum must be selected in the **setGeodeticDatum** command.

Example

```
COM1> sud, User1, 52.1, 49.3, -58.5, 0.891, 5.390, -8.712,
      1.34<CR>
$R: sud, User1, 52.1, 49.3, -58.5, 0.891, 5.390, -8.712, 1.34
      UserDatum, User1, 52.10, 49.30, -58.50, 0.8910, 5.3900, -8.7120,
      1.34000
COM1>
```

sudv gudv	setUserDatumVel getUserDatumVel	Datum Datum	<i>TxVel</i>	<i>TyVel</i>	<i>TzVel</i>	<i>RxVel</i>	<i>RyVel</i>	<i>RzVel</i>	<i>DVel</i>	<i>RefYear</i>
		+ User1 + User2 all	-2000.00 ... 0.00 ... 2000.00 mm/yr	-2000.00 ... 0.00 ... 2000.00 mm/yr	-2000.00 ... 0.00 ... 2000.00 mm/yr	-10.0000 ... 0.0000 ... 10.0000 mas/yr	-10.0000 ... 0.0000 ... 10.0000 mas/yr	-10.0000 ... 0.0000 ... 10.0000 mas/yr	-1.00000 ... 0.00000 ... 1.00000 ppb/yr	1900.00 ... 2000.00 ... 2100.00 yr

[RxControl: Navigation > Receiver Operation > Position > Datum](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Position > Datum](#)

Use these commands to define the time derivative of the seven datum transformation parameters defined with the **setUserDatum** command.

For instance, *TxVel* is the yearly change of the X-translation component. At the epoch specified with *RefYear* (in decimal years), the X-translation component is *Tx* as defined in **setUserDatum**. One year later, the X-translation component is $Tx + TxVel$, etc.

Refer to **setUserDatum** command for a description of the datum transformation formula implemented in the receiver.

Example

```
COM1> sudv, User1, 0.1, 0.1, -1.8, 0.081, 0.49, -0.792, 0.08,
      2000<CR>
$R: sudv, User1, 0.1, 0.1, -1.8, 0.081, 0.49, -0.792, 0.08, 2000
      UserDatumVel, User1, 0.10, 0.10, -1.80, 0.0810, 0.4900, -0.7920,
      0.08000, 2000.00
COM1>
```

sue gue	setUserEllipsoid getUserEllipsoid	Datum Datum	a	Invf						
		+ User1 + User2 all	6300000.000 ... 6378137.000 ... 6400000.000 m	290.000000000 ... 298.257223563 ... 305.000000000						

RxControl: Navigation > Receiver Operation > Position > Datum

Web Interface: Configuration > Navigation > Receiver Operation > Position > Datum

Use these commands to define the ellipsoid associated with the `User1` or `User2` datums.

`a` is the reference ellipsoid semi-major axis and `Invf` is the inverse of the flattening.

See also the `setGeodeticDatum` and the `setUserDatum` commands.

Example

```
COM1> sue, User1, 6378388, 297 <CR>
$R: sue, User1, 6378388, 297
    UserEllipsoid, User1, 6378388.000, 297.000000000
COM1>
```

2.3.5 Receiver Operation Commands

scst gcst	setClockSyncThreshold getClockSyncThreshold	Threshold								
		ClockSteering								
		usec500								
		msec1								
		msec2								
		msec3								
		msec4								
		msec5								

[RxControl: Navigation > Receiver Operation > Timing](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Timing](#)

Use these commands to define/inquire the maximum allowed offset between the receiver internal clock and the system time defined by the **setTimingSystem** command.

If the argument `ClockSteering` is selected, the receiver internal clock is continuously steered to the system time to within a couple of nanoseconds. Clock steering accuracy is dependent on the satellite visibility, and it is recommended to only enable it under open-sky conditions.

If any other argument is selected, the internal clock is left free running, and synchronization with the system time is done through regular millisecond clock jumps. More specifically, when the receiver detects that the time offset is larger than *Threshold*, it initiates a clock jump of an integer number of milliseconds to re-synchronise its internal clock with the system time. These clock jumps have no influence on the generation of the xPPS pulses: the xPPS pulses are always maintained within a few nanoseconds from the requested time, regardless of the value of the *Threshold* argument.

Please refer to the Firmware User Manual for a more detailed description of the time keeping in your receiver.

Example

To enable clock steering, use:

```
COM1> scst, clocksteering <CR>
$R: scst, clocksteering
      ClockSyncThreshold, ClockSteering
COM1>
```


sep gep	setEventParameters getEventParameters	Event Event	Polarity	Delay						
		+ EventA + EventB all	Low2High High2Low	-500.000000 ... 0.000000 ... 500.000000 msec						

RxControl: Navigation > Receiver Operation > Timing

Web Interface: Configuration > Navigation > Receiver Operation > Timing

Use these commands to define/inquire the polarity of the electrical transition on which the receiver will react on its `Event` input(s). The polarity of each event pin can be set individually or simultaneously by using the value `all` for the *Event* argument.

The command also allows defining a time delay for each event. This can be handy when the electrical transition at the event pin is not synchronous with the actual event that needs to be timed. For example, if the electrical transition occurs 100 milliseconds prior to the actual event of interest, the *Delay* argument must be set to 100. *Delay* is positive when the event of interest occurs after the electrical transition, and negative otherwise.

The event time (corrected by the specified delay) is available in the `ExtEvent` SBF block, and the position at that (corrected) time is available in the `ExtEventPVTCartesian` and `ExtEventPVTGeodetic` SBF blocks. Beware that, when using large *Delay* values in high-dynamics conditions, the position accuracy may degrade.

Example

```
COM1> sep, EventA, High2Low, 100 <CR>
$R: sep, EventA, High2Low, 100
      EventParameters, EventA, High2Low, 100.000000
COM1>
```

sgpf ggpf	setGPIOFunctionality getGPIOFunctionality	GPPin GPPin	Mode	Input	Output					
		+ GP1 + GP2 + GP3 all	Output	none	LevelLow LevelHigh					

[RxControl: Navigation > Receiver Operation > GPIO](#)

[Web Interface: Configuration > Navigation > Receiver Operation > GPIO](#)

Use these commands to define/inquire the functionality assigned to every GPIO pin.

Currently, only the output pins (GP_x) can be controlled by this command, and the *Mode* and *Input* arguments can only take the values `Output` and `none` respectively. The argument *Output* sets the electrical level to be applied to the pin specified in *GPPin*.

In housed products, the number of GPIO pins configurable by this command is larger than the number of GPIO pins available to the user. The extra pins are used for internal purposes, and their settings should not be modified. Please refer to the Hardware Manual of your product to check which GPIO pins are available.

Example

To set the signal on GP2 to a logical 1, use:

```
COM1> sgpf, GP2, Output, , LevelHigh <CR>
$R: sgpf, GP2, Output, , LevelHigh
    GPIOFunctionality, GP2, Output, none, LevelHigh
COM1>
```

slm	setLEDMode	GPLED								
glm	getLEDMode	DIFFCORLED								
		PVTLED								
		LOGLED								

[RxControl: Navigation > Receiver Operation > GPIO](#)

[Web Interface: Configuration > Navigation > Receiver Operation > GPIO](#)

Use these commands to define/inquire the blinking mode of the General Purpose LED (GPLED).

The different LED blinking modes are described in the Hardware Manual of your receiver.

Example

```
COM1> slm, DIFFCORLED <CR>
$R: slm, DIFFCORLED
      LEDMode, DIFFCORLED
COM1>
```

spps gpps	setPPSParameters getPPSParameters	Interval	Polarity	Delay	TimeScale	MaxSyncAge				
		off msec100 msec200 msec500 sec1 sec2 sec5 sec10	Low2High High2Low	-1000000.00 ... 0.00 ... 1000000.00 nsec	TimeSys UTC RxClock GLONASS	1 ... 60 ... 3600 sec				

[RxControl: Navigation > Receiver Operation > Timing](#)

[Web Interface: Configuration > Navigation > Receiver Operation > Timing](#)

Use these commands to define/inquire the interval, the polarity, the delay and time scale of the x-pulse-per-second (xPPS) output. Please refer to the Firmware User Manual for a detailed description of the xPPS functionality.

The *Interval* argument specifies the time interval between the pulses. A special value "off" is defined to disable the xPPS signal.

The *Polarity* argument defines the polarity of the xPPS signal.

The *Delay* argument can be used to compensate for the overall signal delays in the system (including antenna, antenna cable and xPPS cable). Setting *Delay* to a higher value causes the xPPS pulse to be generated earlier. For example, if the antenna cable is replaced by a longer one, the overall signal delay could be increased by, say, 20 nsec. If *Delay* is left unchanged, the xPPS pulse will come 20 nsec too late. To re-synchronize the xPPS pulse, *Delay* has to be increased by 20 nsec.

By default, the xPPS pulses are aligned with the satellite time system (*TimeSys*) that is defined by the **setTimingSystem** command. Using the *TimeScale* argument, it is also possible to align the xPPS pulse with the local receiver time (*RxClock*), with GLONASS time or with UTC.

When *TimeScale* is set to anything else than *RxClock*, the accuracy of the position of the xPPS pulse depends on the age of the last PVT computation. During PVT outages (due for instance to signal blockage), the xPPS position is extrapolated on the basis of the last available PVT information, and may start to drift. To avoid large biases, the receiver stops outputting the xPPS pulse when the last PVT is older than the age specified in the *MaxSyncAge* argument. *MaxSyncAge* is ignored when *TimeScale* is set to *RxClock*.

Examples

```
COM1> spps, sec1, , 23.4 <CR>
$R: spps, sec1, , 23.4
    PPSParameters, sec1, Low2High, 23.40, TimeSys, 60
COM1>

COM1> gpps <CR>
$R: gpps
    PPSParameters, sec1, Low2High, 23.40, TimeSys, 60
COM1>
```

swui gwui	setWakeUpInterval getWakeUpInterval	WakeUpTime (30)	AwakeDuration	RepetitionPeriod						
		2000-01-01 00:00:00	0 ... 604800 sec	0 ... 604800 sec						

RxControl: [File > Power Mode > Scheduling](#)

Web Interface: [Receiver > Administration > Power Mode > Scheduling](#)

This command can be used to set up an automatic receiver awake/sleep pattern. It is possible to order the receiver to awake at a given time, for a certain period, and/or at regular intervals. A possible application is keeping fast time-to-first-fix even after days in sleep mode. This can be done by waking up the receiver every few hours for a few minutes, such that it can regularly refresh its ephemerides.

The *WakeUpTime* argument defines the epoch when the receiver should automatically wake up the first time. It also serves as reference epoch for the *RepetitionPeriod* argument. The time system in which the user should express the *WakeUpTime* is the one defined by the **setTimingSystem** command. The format of the *WakeUpTime* argument is "YYYY-MM-DD hh:mm:ss".

The *AwakeDuration* argument defines the period for which the receiver should stay awake. If this argument is set to 0 (the default value), the receiver will remain awake indefinitely.

The *RepetitionPeriod* can be used to repeat the awake/sleep pattern at regular interval. *RepetitionPeriod* should be at least 5 seconds longer than *AwakeDuration* to allow a minimum sleep time of 5 seconds between awake periods. If *RepetitionPeriod* is set to a value smaller than *AwakeDuration*, the repetition functionality is disabled.

Be aware that the receiver must know the time to automatically go into sleep mode, which requires signal tracking: if no antenna is connected to the receiver or if no satellite can be tracked during the *AwakeDuration*, the receiver will continue operating beyond its prescribed awake duration, and only possibly enter sleep mode at the next scheduled "go-to-sleep" epoch, if any.

To force the receiver to go into sleep mode immediately, use the command **exePowerMode, ScheduledSleep** instead.

If interaction with the receiver is needed during the sleep period, the user can always force the receiver to wake up by hardware means. The different ways to do so are described in the Hardware Manual. Usually, simply sending any character at the correct baud rate to the COM1 port wakes up the receiver. When maintenance is done, the user should put the receiver back in sleep mode by typing **exePowerMode, ScheduledSleep**. This does not perturb the awake/sleep pattern: the receiver will continue to automatically wake up at the next wake-up epoch.

Examples

If you want the receiver waking up on December 31, 2012 at 23h00 for 2 hours, use:

```
COM1> swui, "2012-12-31 23:0:0", 7200 <CR>
$R: swui, "2012-12-31 23:0:0", 7200
WakeUpInterval, "2012-12-31 23:00:00", 7200, 0
COM1>
```

If you want to set up an automatic wake up every day at midnight for 1 hour, use:

```
COM1> swui, , 3600, 86400 <CR>
```

```
$R: swui, , 3600, 86400
    WakeUpInterval, "2000-01-01 00:00:00", 3600, 86400
COM1>
```

2.3.6 Session Settings Commands

smp gmp	setMarkerParameters getMarkerParameters	MarkerName (60)	MarkerNumber (2)	MarkerType (20)						
		SEPT	Unknown	Unknown						

[RxControl: Navigation > Receiver Setup > Station Settings](#)

[Web Interface: Configuration > Navigation > Receiver Setup > Station Settings](#)

Use these commands to define/inquire the marker parameters.

The set of allowed characters for the *MarkerName* argument is:

_0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

If internal SBF or NMEA logging is enabled in one of the `IGS` file naming modes (see **setFileNaming** command), the file name is determined by the *MarkerName* argument. Changing *MarkerName* will cause the current log file to be closed, and a new file to be created.

When generating a RINEX observation file with the `sbf2rin` utility, the marker parameters are reflected in the header section and a "new site occupation" event is inserted between observation records each time the marker name or number is changed with this command.

Example

```
COM1> smp, Test, 356, GEODETIC<CR>
$R: smp, Test, 356, GEODETIC
    MarkerParameters, Test, 356, GEODETIC
COM1>
```


soc	setObserverComment	Comment (120)								
goc	getObserverComment	Unknown								

[RxControl: Navigation > Receiver Setup > Station Settings](#)

[Web Interface: Configuration > Navigation > Receiver Setup > Station Settings](#)

Use these commands to define/inquire the content of the `Comment` SBF block.

Examples

```
COM1> soc, "Data taken with choke ring antenna" <CR>
$R: soc, "Data taken with choke ring antenna"
      ObserverComment, "Data taken with choke ring antenna"
COM1>

COM1> goc <CR>
$R: goc
      ObserverComment, "Data taken with choke ring antenna"
COM1>
```

sop	setObserverParameters	Observer (20)	Agency (40)							
gop	getObserverParameters	Unknown	Unknown							

RxControl: Navigation > Receiver Setup > Station Settings

Web Interface: Configuration > Navigation > Receiver Setup > Station Settings

Use these commands to define/inquire the observer name or ID, and his/her agency. These parameters are copied in the `ReceiverSetup` SBF block and in the header of RINEX observation files.

The length of the arguments complies with the RINEX format definition.

Examples

```
COM1>  sop, TestObserver, TestAgency <CR>
$R: sop, TestObserver, TestAgency
      ObserverParameters, "TestObserver", "TestAgency"
COM1>

COM1>  gop <CR>
$R: gop
      ObserverParameters, "TestObserver", "TestAgency"
COM1>
```

2.3.7 Input/Output Commands

scs gcs	setCOMSettings getCOMSettings	Cd Cd	Rate	DataBits	Parity	StopBits	FlowControl			
		+COM1 +COM2 +COM3 +COM4 all	baud1200 baud2400 baud4800 baud9600 baud19200 baud38400 baud57600 <u>baud115200</u> baud230400 baud460800	bits8	No	bit1	none RTS CTS			

[RxControl: Communication > COM Port Settings](#)

[Web Interface: Configuration > Communication > COM Port Settings](#)

Use these commands to define/inquire the communication settings of the receiver's COM ports. By default, all COM ports are set to a baud rate of 115200 baud, using 8 data-bits, no parity, 1 stop-bit and no flow control.

Depending on your receiver hardware, it may be that not all COM ports support flow control. Please refer to the Hardware Manual to check which COM ports are equipped with the RTS/CTS lines.

In some housed products, the number of COM ports configurable by this command is larger than the number of COM ports available to the user. The extra COM ports are used for internal purposes, and their settings should not be modified. Please refer to the Hardware Manual of your product for further details.

When modifying the settings of the current connection, make sure to also modify the settings of your terminal emulation program accordingly.

Example

```
COM1> scs, COM1, baud19200, bits8, No, bit1, RTS|CTS<CR>
$R: scs, COM1, baud19200, bits8, No, bit1, RTS|CTS
    COMSettings, COM1, baud19200, bits8, No, bit1, RTS|CTS
COM1>
```

scda	setCrossDomainWebAccess	Mode								
gcda	getCrossDomainWebAccess									
		off								
		on								

[RxControl: Communication > Network Settings](#)

[Web Interface: Configuration > Communication > Network Settings](#)

This command enables or disables true open access across domain boundaries according to the CORS specification (Cross-Origin Resource Sharing).

Setting the *Mode* argument to `on` enables the cross-domain access to the receiver web server, and as such it allows external client applications (e.g. your own web application) to access receiver data via HTTP requests. Please contact support for additional information on the receiver's JavaScript libraries.

Example

```
COM1> scda, on <CR>
$R: scda, on
      CrossDomainWebAccess, on
COM1>
```

sdio gdio	setDataInOut getDataInOut	Cd Cd	Input	Output	Show					
		+ DSK1	none	none	(off)					
		+ COM1	CMD	+ RTCMv2	(on)					
		+ COM2	RTCMv2	+ RTCMv3						
		+ COM3	RTCMv3	+ CMRv2						
		+ COM4	CMRv2	+ SBF						
		+ USB1	DC1	+ NMEA						
		+ USB2	DC2	+ ASCIIIDisplay						
		+ IP10 ... IP17	RTCMV	+ DC1						
		+ NTR1	ASCIIIN	+ DC2						
		+ NTR2	auto							
		+ NTR3								
		+ IPS1								
		+ IPS2								
		+ IPS3								
		+ IPR1								
		+ IPR2								
		+ IPR3								
		all								

RxControl: [Communication > Input/Output Selection](#)

Web Interface: [Configuration > Communication > Input/Output Selection](#)

Use these commands to define/inquire the type of data that the receiver should accept/send on a given connection descriptor (*Cd*).

The *Input* argument is used to tell the receiver how to interpret incoming bytes on the connection *Cd*. If a connection is to be used for receiving user commands or differential corrections in RTCM or CMR format, it is recommended to leave it in the default `auto` input mode. In this mode, the receiver automatically detects the input format.

It is also possible to set the input format explicitly. `CMD` means that the connection is to be used for user command input exclusively. `RTCMv2`, `RTCMv3` and `CMRv2` can be used to manually select the differential correction format, overriding the auto detection. `RTCMV` is the LBAS1-proprietary differential correction stream decoded from L-Band. `ASCIIIN` is used for connections receiving free-formatted ASCII messages, e.g. from an external meteo sensor.

In `auto` mode, the receiver automatically detects the `CMD`, `RTCMv2`, `RTCMv3`, `RTCMV` or `CMRv2` formats. The other input formats must be specified explicitly.

A connection that is not configured in `CMD` mode or `auto` mode will be blocked for user commands. There are two ways to re-enable the command input on a blocked connection. The first way is to reconfigure the connection by entering the command `setDataInOut` from another connection. The second way is to send the "escape sequence" consisting of a succession of ten "S" characters to the blocked connection within a time interval shorter than 5 seconds.

A connection that is configured in `auto` mode will initially accept user commands and differential corrections. However, as soon as differential corrections have been detected, the connection is blocked for user commands until the escape sequence is received.

The *Output* argument is used to select the types of data allowed as output. The receiver supports outputting different data types on the same connection. The `ASCIIIDisplay` is a textual report of the tracking and PVT status at a fixed rate of 1Hz. It can be used to get a quick overview of the receiver operation.

DC1 and DC2 represent two internal pipes that can be used to create a daisy-chain. Set the *Input* argument to DC*i* to connect the input of pipe *i* to the specified connection. Set the *Output* argument to DC*i* to connect the output of pipe *i* to the specified connection.

After the *Cd*, *Input* and *Output* arguments, an extra read-only *Show* argument will be returned in the command reply. This last argument can take the value `on` or `off`, depending on whether the connection descriptor is open or not.

The *Input* argument is ignored for output-only connections, such as DSK1 or IPSx.

The *Output* argument is ignored for the IP receive (IPRx) connections. In addition, IPRx connections cannot be used to enter user commands.

If a NTRIP connection (NTRx) is configured as Server with the command **setNTRIPSettings**, the *Input* argument for that connection is ignored.

By default, pressing the log button (or, in OEM receivers, driving the Button pin low) has the effect of executing the commands **sdio,DSK1,,SBF+NMEA** and **sdio,DSK1,,none** in turn: it toggles internal SBF and NMEA logging on and off (pressing the log button has no effect on the internal RINEX logging).

Examples

```
COM1> sdio, COM2, RTCMv2 <CR>
$R: sdio, COM2, RTCMv2
    DataInOut, COM2, RTCMv2, SBF+NMEA, (on)
COM1>
```

To setup a two-way daisy-chain between COM1 and COM2:

```
COM1> sdio, COM1, DC1, DC2 <CR>
$R: sdio, COM1, DC1, DC2
    DataInOut, COM1, DC1, DC2, (on)
COM1> sdio, COM2, DC2, DC1 <CR>
$R: sdio, COM2, DC2, DC1
    DataInOut, COM2, DC2, DC1, (on)
COM1>
```

eecm gecm	exeEchoMessage getEchoMessage	Cd	Message (242)	EndOfLine						
		DSK1 COM1 COM2 COM3 COM4 USB1 USB2 IP10 ... IP17	A:Unknown	none + CR + LF all						

[RxControl: Communication > Output Settings > Echo Message](#)

[Web Interface: Configuration > Communication > Output Settings > Echo Message](#)

Use this command to send a message to one of the connections of the receiver.

The *Message* argument defines the message that should be sent on the *Cd* port. If the given message starts with "A:", the remainder of the message is considered an ASCII string that will be forwarded without changes to the requested connection. If the given message starts with "H:", the remainder of the message is considered a hexadecimal representation of a succession of bytes to be sent to the requested connection. In this case, the string should be a succession of 2-character hexadecimal values separated by a single whitespace.

Make sure to enclose the string between double quotes if it contains whitespaces. The maximum length of the *Message* argument (including the A: or H: prefix) is 242 characters.

The *EndOfLine* argument defines which end-of-line character should be sent after the message. That argument is ignored when the *Message* argument starts with H:.

To send a message at a regular interval instead of once, use the command **setPeriodicEcho**.

Examples

To send the string "Hello world!" to COM2, use:

```
COM1> eecm, COM2, "A:Hello world!", none <CR>
$R: eecm, COM2, "A:Hello world!", none
EchoMessage, COM2, "A:Hello world!", none
COM1>
```

To send the same string, the following command can also be used:

```
COM1> eecm, COM2, "H:48 65 6C 6C 6F 20 77 6F 72 6C 64 21", none
<CR>
$R: eecm, COM2, "H:48 65 6C 6C 6F 20 77 6F 72 6C 64 21", none
EchoMessage, COM2, "H:48 65 6C 6C 6F 20 77 6F 72 6C 64 21", none
COM1>
```


sipp	setIPPortSettings	Command								
gipp	getIPPortSettings									
		1 ... 28784 ... 65535								

RxControl: Communication > Network Settings

Web Interface: Configuration > Communication > Network Settings

Use these commands to define/inquire the port number where the receiver listens for incoming TCP/IP connections.

For the new setting to become active, you need to reset the receiver (e.g. by the command **exeResetReceiver, soft, none**).

The IP port number configured by this command keeps its value upon a power cycle and even after a reset to factory default (see command **exeResetReceiver**).

Note that this command is not shown in the output of the **1stConfigFile** command.

Example

```
COM1> sipp, 12345 <CR>
$R: sipp, 12345
    IPPortSettings, 12345
COM1>
```

sirs girs	setIPReceiveSettings getIPReceiveSettings	Cd Cd	Port	Mode	TCPAddress (40)					
		+ IPR1 + IPR2 + IPR3 all	0 ... 65535	TCP UDP	0.0.0.0					

RxControl: Communication > Network Settings

Web Interface: Configuration > Communication > Network Settings

This command configures the "IP receive" ports (IPR).

When *Mode* is set to `TCP`, the receiver connects to the specified port of a server of which the IP address or hostname is provided in the *TCPAddress* argument. It then receives all data sent by this server on that port.

When *Mode* is set to `UDP`, the receiver listens for incoming UDP messages on its port identified by the *Port* argument. In `UDP` mode, the *TCPAddress* argument is ignored.

If *Port* is set to 0, the corresponding IPR connection is disabled.

`IPRx` connections are typically used for differential correction input. It is not possible to enter user commands through `IPRx` connections.

If there is no incoming data for more than 10 seconds, the receiver closes the connection and tries to reconnect.

Note that this command is the counterpart of the `setIPServerSettings` command. `setIPServerSettings` configures the sender side of the communication, while `setIPReceiveSettings` configures the receiver side.

Example

```
COM1> sirs, IPR1, 28785, TCP, 192.168.10.5<CR>
$R: sirs, IPR1, 28785, TCP, 192.168.10.5
    IPReceiveSettings, IPR1, 28785, TCP, 192.168.10.5
COM1>
```

sisss giss	setIPServerSettings getIPServerSettings	Cd Cd	Port	Mode	UDPAddress (200)					
		+ IPS1 + IPS2 + IPS3 all	0 ... 65535	TCP UDP	255.255.255.255					

RxControl: [Communication > Network Settings](#)

Web Interface: [Configuration > Communication > Network Settings](#)

By default (*Mode* set to `TCP`), this command defines the TCP/IP port where the receiver's IP Servers (IPS) listen for incoming TCP/IP connections. When a client connects to an IPS port, all output data specified for that port are streamed to the client.

When *Mode* is set to `UDP` and *UDPAddress* is set to `255.255.255.255`, the IPS works in UDP broadcast mode. In that mode, the IPS data stream is delivered to any host on the local network listening to the IP port specified by the *Port* argument.

When *Mode* is set to `UDP` and *UDPAddress* contains a whitespace-separated list of IP addresses or hostnames, the IPS data stream is only delivered to the specified hosts.

Use the `setDataInOut` command and the various output setting commands (e.g. `setNMEAOutput`) to define the data stream to be output by the IPS connections. Note that the UDP implementation is meant to be used with small data volumes and low update rates. It is the user's responsibility to only enable short messages at low rate when using UDP, in order to prevent throughput degradation of the network.

It is possible to configure some IPS connections in UDP mode, and others in TCP mode. The *UDPAddress* argument is ignored in TCP mode.

Set the *Port* argument to 0 to disable an IPS connection.

For the new setting to become active, you need to reset the receiver (e.g. by the command `exeResetReceiver, soft, none`). To make the setting persistent, it must be saved in the boot configuration file with the command `exeCopyConfigFile`.

Example

```
COM1> sisss, IPS1, 28785, UDP, 255.255.255.255<CR>
$R: sisss, IPS1, 28785, UDP, 255.255.255.255
    IPServerSettings, IPS1, 28785, UDP, 255.255.255.255
COM1>
```

sips gips	setIPSettings getIPSettings	Mode	IP (16)	Netmask (16)	Gateway (16)	Domain (63)	DNS1 (16)	DNS2 (16)		
		DHCP	192.168.2.2	255.255.255.0	192.168.2.1		8.8.8.8	8.8.4.4		
		Static								

[RxControl: Communication > Network Settings](#)

[Web Interface: Configuration > Communication > Network Settings](#)

Use these commands to define the IP (Internet Protocol) settings of the receiver's Ethernet port. By default, the receiver is configured to use DHCP.

In `Static` mode, the receiver will not attempt to request an address via DHCP. It will use the specified IP address, netmask, gateway, domain name and DNS. `DNS1` is the primary DNS, and `DNS2` is the backup DNS. The arguments `IP`, `Netmask`, `Gateway`, `Domain`, `DNS1`, and `DNS2` are ignored in `DHCP` mode.

For the new settings to become active, you need to reset the receiver (e.g. by the command **exeResetReceiver**, **soft**, **none**).

The IP settings configured by this command keep their value upon a power cycle and even after a reset to factory default (see command **exeResetReceiver**).

Note that this command is not shown in the output of the **1stConfigFile** command.

The command **getIPSettings** cannot be used to get the current IP address assigned to the receiver by the DHCP server. The current IP address can be retrieved from the command **1stInternalFile**, **IPParameters**, or from the `IPStatus` SBF block.

Example

```
COM1> sips, static, 192.168.1.123, 255.255.252.0, 192.168.1.255,
      domain.local, 192.168.100.3, 192.168.100.4 <CR>
$R: sips, static, 192.168.1.123, 255.255.252.0, 192.168.1.255,
    domain.local, 192.168.100.3, 192.168.100.4
    IPSettings, Static, 192.168.1.123, 255.255.252.0, 192.168.1.255,
    domain.local, 192.168.100.3, 192.168.100.4
COM1>
```

enoc gnoc	exeNMEAOnce getNMEAOnce	Cd	Messages							
		DSK1	+ALM							
		COM1	+DTM							
		COM2	+GBS							
		COM3	+GGA							
		COM4	+GLL							
		USB1	+GNS							
		USB2	+GRS							
		IP10 ... IP17	+GSA							
		NTR1	+GST							
		NTR2	+GSV							
		NTR3	+HDT							
		IPS1	+RMC							
		IPS2	+ROT							
		IPS3	+VTG							
			+ZDA							
			+HRP							
			+LLQ							
			+RBP							
			+RBV							
			+RBD							
			+AVR							
			+GGK							
			+GFA							
			+GGQ							
			+LLK							
			+GMP							
			+TXTbase							

RxControl: Communication > Output Settings > NMEA Output Once

Web Interface: Configuration > Communication > Output Settings > NMEA Output Once

Use this command to output a set of NMEA messages [2] on a given connection. This command differs from the related **setNMEAOutput** command in that it instructs the receiver to output the specified messages only once, instead of at regular intervals.

The *Cd* argument defines the connection descriptor on which the message(s) should be output and the *Messages* argument defines the list of messages that should be output. The HRP, RBP, RBD and RBV messages are non-standard. They are described in the Firmware User Manual. TXTbase is a TXT message containing text transmitted by a base station in RTCM message type 1029.

Please make sure that the connection specified by *Cd* is configured to allow NMEA output (this is the default for all connections). See the **setDataInOut** command.

Example

To output the receiver position on COM1, use:

```
COM1> enoc, COM1, GGA <CR>
$R: enoc, COM1, GGA
NMEAOnce, COM1, GGA
COM1>
```

sno gno	setNMEAOutput getNMEAOutput	Stream Stream	Cd	Messages	Interval					
		+ Stream1 ... Stream10 all	none DSK1 COM1 COM2 COM3 COM4 USB1 USB2 IP10 ... IP17 NTR1 NTR2 NTR3 IPS1 IPS2 IPS3	none + ALM + DTM + GBS + GGA + GLL + GNS + GRS + GSA + GST + GSV + HDT + RMC + ROT + VTG + ZDA + HRP + LLQ + RBP + RBV + RBD + PUMRD + AVR + GGK + GFA + GGQ + LLK + GMP + TXTbase	off OnChange msec10 msec20 msec40 msec50 msec100 msec200 msec500 sec1 sec2 sec5 sec10 sec15 sec30 sec60 min2 min5 min10 min15 min30 min60					

[RxControl: Communication > Output Settings > NMEA Output > NMEA Output Intervals](#)

[Web Interface: Configuration > Communication > Output Settings > NMEA Output > NMEA Output Intervals](#)

Use this command to output a set of NMEA messages [2] on a given connection at a regular interval. The *Cd* argument defines the connection descriptor on which the message(s) should be output and the *Messages* argument defines the list of messages that should be output. The HRP, RBP, RBD, RBV and PUMRD messages are non-standard. They are described in the Firmware User Manual. TXTbase is a TXT message containing text transmitted by a base station in RTCM message type 1029.

This command is the counterpart of the **setSBFOutput** command for NMEA sentences. Please refer to the description of that command for a description of the arguments.

Examples

To output GGA at 1Hz and RMC at 10Hz on COM1, use:

```
COM1> sno, Stream1, COM1, GGA, sec1 <CR>
$R: sno, Stream1, COM1, GGA, sec1
NMEAOutput, Stream1, COM1, GGA, sec1
COM1> sno, Stream2, COM1, RMC, msec100 <CR>
$R: sno, Stream2, COM1, RMC, msec100
NMEAOutput, Stream2, COM1, RMC, msec100
COM1>
```

To get the list of NMEA messages currently output, use:

```
COM1> gno <CR>
$R: gno
NMEAOutput, Stream1, COM1, GGA, sec1
```

```
NMEAOutput, Stream2, COM1, RMC, msec100
NMEAOutput, Stream3, none, none, off
NMEAOutput, Stream4, none, none, off
NMEAOutput, Stream5, none, none, off
NMEAOutput, Stream6, none, none, off
NMEAOutput, Stream7, none, none, off
NMEAOutput, Stream8, none, none, off
NMEAOutput, Stream9, none, none, off
NMEAOutput, Stream10, none, none, off
COM1>
```


snp gnp	setNMEAPrecision getNMEAPrecision	NrExtraDigits	Compatibility	LocalDatum						
		0 ... 3	Nominal Mode1 Mode2	off only						

[RxControl: Communication > Output Settings > NMEA Output > Customize](#)

[Web Interface: Configuration > Communication > Output Settings > NMEA Output > Customize](#)

Use these commands to define/inquire the number of extra digits in the latitude, longitude and altitude reported in NMEA sentences and to tune certain sentences to be compatible with third-party applications that are not fully compliant with the NMEA 0183 standard.

By default (*NrExtraDigits* is 0), latitude and longitude are reported in degrees with 5 decimal digit, and altitude is reported in meters with 2 decimal digit. These default numbers of digits lead to a centimeter-level resolution of the position. To represent RTK positions with their full precision (millimeter-level), it is recommended to set *NrExtraDigits* to 2.

It is important to note that increasing the number of digits (setting *NrExtraDigits* to a non-zero value) may cause the NMEA standard to be broken, as the total number of characters in a sentence may end up exceeding the prescribed limit of 82. This is why it is not done by default.

When setting the argument *Compatibility* to *Mode1*, the GPS Quality Indicator in GGA sentences is set to the value "2: Differential GPS" for all non-standalone positioning modes, the Mode Indicator in GNS sentences is set to "D: Differential" for all non-standalone positioning modes, and the Course Over Ground in the VTG sentences is not a null field for stationary receivers.

When setting the argument *Compatibility* to *Mode2*, the Course Over Ground in the VTG sentences is not a null field for stationary receivers.

The *LocalDatum* argument specifies whether transformation parameters sent out by the RTK service provider should be applied or not in NMEA sentences GGA and GNS. If *LocalDatum* is *off*, the transformation parameters are not applied, and the coordinates in GGA and GNS correspond to the coordinates in the *PVTGeodetic* SBF block. If *LocalDatum* is *only*, the coordinates are transformed to the local datum and correspond to the *PosLocal* SBF block. In that mode, no position is output until the relevant transformation parameters are received in the differential correction stream.

Examples

```
COM1> snp, 2 <CR>
$R: snp, 2
      NMEAPrecision, 2, Nominal, off
COM1>
```

```
COM1> gnp <CR>
$R: gnp
      NMEAPrecision, 2, Nominal, off
COM1>
```

snti	setNMEATalkerID	TalkerID								
gnti	getNMEATalkerID	GP								
		GN								

RxControl: Communication > Output Settings > NMEA Output > Customize

Web Interface: Configuration > Communication > Output Settings > NMEA Output > Customize

Use these commands to define/inquire the "Device Talker" for NMEA sentences. The device talker allows users to identify the type of equipment from which the NMEA sentence was issued.

Note that the command is ignored for the NMEA sentences where it would conflict with the standard. For example, the GSV sentence reporting the GPS visibility will always have its device talker set to "GP" regardless of the **setNMEATalkerID** command.

Example

```
COM1> snti, GN <CR>
$R: snti, GN
      NMEATalkerID, GN
COM1>
```

snv	setNMEAVersion	Version								
gnv	getNMEAVersion									
		v3x								
		v4x								

RxControl: Communication > Output Settings > NMEA Output > Customize

Web Interface: Configuration > Communication > Output Settings > NMEA Output > Customize

Use this command to set the NMEA version the receiver should comply with.

Example

```
COM1> snv, v4x<CR>
$R: snv, v4x
    NMEAVersion, v4x
COM1>
```

snts gnts	setNtripSettings getNtripSettings	Cd Cd	Mode	Caster (40)	Port	UserName (20)	Password (40)	MountPoint (32)	Version	SendGGA
		+NTR1 +NTR2 +NTR3 all	off Server Client		0...2101 ...65535				v1 v2	auto off sec1 sec5 sec10 sec60

RxControl: [Communication > NTRIP Settings](#)

Web Interface: [Configuration > Communication > NTRIP Settings](#)

Use this command to specify the parameters of the NTRIP connection referenced by the *Cd* argument.

The *Mode* argument specifies the type of NTRIP connection. In *Server* mode, the receiver is sending data to a NTRIP caster. In *Client* mode, the receiver gets data from the NTRIP caster. Set *Mode* to *off* to disable the connection.

Caster is the hostname or IP address of the NTRIP caster to connect to. *Port*, *UserName*, *Password* and *MountPoint* are the IP port number, the user name, the password and the mount point to be used when connecting to the NTRIP caster. The default NTRIP port number is 2101. Note that the receiver encrypts the password so that it cannot be read back with the command **getNtripSettings**.

The *Version* argument specifies which version of the NTRIP protocol to use (v1 or v2).

The *SendGGA* argument specifies whether or not to send NMEA GGA messages to the NTRIP caster, and at which rate. In *auto* mode (the default), the receiver automatically sends GGA messages if requested by the caster. This argument is ignored in NTRIP server mode.

Example

```
COM1> snts, NTR1, Client, ntrip.com, 2101, USER, PWD, MP1, v2,
      auto<CR>
$R: snts, NTR1, Client, ntrip.com, 2101, USER, PWD, MP1, v2, auto
    NtripSettings, NTR1, Client, ntrip.com, 2101, USER, PWD, MP1, v2,
    auto
COM1>
```

Inst	InstNTRIPSourceTable	Caster (40)	Port							
			0...2101 ...65535							

Use this command to retrieve the source table from the specified NTRIP caster.

Caster is the hostname or IP address of the NTRIP caster to connect to, and *Port* is the IP port number. The default NTRIP port number is 2101.

Example

```
COM1> inst, ntripcaster <CR>
$R; inst, ntripcaster
---->
$-- BLOCK 1 / 0 C
HTTP/1.1 200 OK
Ntrip-Version: Ntrip/2.0
Ntrip-Flags: st_filter,st_auth,st_match,st_strict,rtsp,plain_rtp
Server: NTRIP Caster/2.0.15
...
$-- BLOCK 1 / 0 C
ENDSOURCETABLE
COM1>
```

spe gpe	setPeriodicEcho getPeriodicEcho	Cd Cd	Message (201)	Interval						
		+ COM1 + COM2 + COM3 + COM4 all	A:Unknown	off once msec100 msec200 msec500 sec1 sec2 sec5 sec10 sec15 sec30 sec60 min2 min5 min10 min15 min30 min60						

[RxControl: Communication > Output Settings > Periodic Echo message](#)

[Web Interface: Configuration > Communication > Output Settings > Periodic Echo message](#)

Use this command to periodically send a message to one of the connections of the receiver.

The *Message* argument defines the message that should be sent on the *Cd* port. If the given message starts with "A:", the remainder of the message is considered an ASCII string that will be forwarded to the requested connection. All occurrences of the %%CR character sequence are replaced by a single carriage return character (ASCII code 13d) and all occurrences of the %%LF character sequence are replaced by a single line feed character (ASCII code 10d). If the *Message* argument starts with "H:", the remainder of the message is considered a hexadecimal representation of a succession of bytes to be sent to the requested connection. In this case, the string should be a succession of 2-character hexadecimal values separated by a single whitespace.

Make sure to enclose the string between double quotes if it contains whitespaces. The maximum length of the *Message* argument (including the A: or H: prefix) is 201 characters.

The *Interval* argument defines the interval at which the message should be sent.

To send a message only once, set *Interval* to `once`. The only difference with the command **exeEchoMessage** is that **exeEchoMessage** cannot be stored in the boot configuration file, while **setPeriodicEcho** can. This can be used to output a message once at each reset or reboot. The third example below shows how to do this.

Examples

To send the string "Hello!<CR><LF>" to COM2 every minute, use:

```
COM1> spe, COM2, "A:Hello!%%CR%%LF", sec60 <CR>
$R: spe, COM2, "A:Hello!%%CR%%LF", sec60
PeriodicEcho, COM2, "A:Hello!%%CR%%LF", sec60
COM1>
```

The same can be achieved with the following command:

```
COM1> spe, COM2, "H:48 65 6C 6C 6F 21 0D 0A", sec60 <CR>
$R: spe, COM2, "H:48 65 6C 6C 6F 21 0D 0A", sec60
PeriodicEcho, COM2, "H:48 65 6C 6C 6F 21 0D 0A", sec60
```

COM1>

To let the receiver output the string "Hello!<CR><LF>" to COM2 at each reset, use the following command sequence:

```
COM1> spe, COM2, "A:Hello!%%CR%%LF", once <CR>
$R: spe, COM2, "A:Hello!%%CR%%LF", once
    PeriodicEcho, COM2, "A:Hello!%%CR%%LF", once
COM1> eccf, Current, Boot <CR>
$R: eccf, Current, Boot
    CopyConfigFile, Current, Boot
COM1>
```


ssgp gsgp	setSBFGroups getSBFGroups	Group Group	Messages							
		+ Group1	none							
		+ Group2	[SBF List]							
		+ Group3	+ Measurements							
		+ Group4	+ RawNavBits							
		all	+ GPS							
			+ GLO							
			+ GAL							
			+ GEO							
			+ CMP							
			+ QZS							
			+ PVTCart							
			+ PVTGeod							
			+ PVTEExtra							
			+ Attitude							
			+ Time							
			+ Events							
			+ DiffCorr							
			+ Status							
			+ LBand							
			+ Rinex							
			+ Support							
			+ RawData							
			+ PostProcess							
			+ GUI							

[RxControl: Communication > Output Settings > SBF Groups](#)

[Web Interface: Configuration > Communication > Output Settings > SBF Groups](#)

Use these commands to define/inquire user-defined groups of SBF blocks that can be re-used in the **exeSBFOnce** and the **setSBFOutput** commands. The purpose of defining groups is to ease the typing effort when the same set of SBF blocks are to be addressed regularly.

The list of supported SBF blocks [SBF List] is to be found in Section 2.3.13, "SBF List".

A number of predefined groups of SBF blocks are available (such as `Measurements` or `RawNavBits`). See the command **setSBFOutput** for a description of these predefined groups.

Example

To output the messages `MeasEpoch`, `PVTCartesian` and `DOP` as one group on COM1 at a rate of 1Hz, you could use the following sequence of commands:

```
COM1> ssgp, Group1, MeasEpoch+PVTCartesian+DOP <CR>
$R: ssgp, Group1, MeasEpoch+PVTCartesian+DOP
      SBFGroups, Group1, MeasEpoch+PVTCartesian+DOP
COM1> sso, Stream1, COM1, Group1, sec1 <CR>
$R: sso, Stream1, COM1, Group1, sec1
      SBFOutput, Stream1, COM1, Group1, sec1
COM1>
```

esoc gsoc	exeSBFOnce getSBFOnce	Cd	Messages							
		DSK1	[SBF List]							
		COM1	+ Measurements							
		COM2	+ GPS							
		COM3	+ GLO							
		COM4	+ GAL							
		USB1	+ GEO							
		USB2	+ CMP							
		IP10 ... IP17	+ QZS							
		NTR1	+ PVTCart							
		NTR2	+ PVTGeod							
		NTR3	+ PVTEExtra							
		IPS1	+ Attitude							
		IPS2	+ Time							
		IPS3	+ Status							
			+ LBand							
			+ UserGroups							
			+ Rinex							
			+ Support							
			+ RawData							
			+ PostProcess							
			+ GUI							

RxControl: [Communication > Output Settings > SBF Output Once](#)

Web Interface: [Configuration > Communication > Output Settings > SBF Output Once](#)

Use this command to output a set of SBF blocks on a given connection. This command differs from the related **setSBFOutput** command in that it instructs the receiver to output the specified SBF blocks only once, instead of at regular intervals.

The *Cd* argument defines the connection descriptor on which the message(s) should be output and the *Messages* argument defines the list of messages that should be output. The list of SBF blocks [SBF List] is to be found in Section 2.3.13, "SBF List". Only a subset of SBF blocks can be sent with the **exeSBFOnce** command: refer to the SBF Reference Guide for a list of them.

Make sure that the connection specified by *Cd* is configured to allow SBF output (this is the default for all connections). See also the **setDataInOut** command.

Predefined groups of SBF blocks (such as `Measurements` or `PVTCart`) can be addressed in the *Messages* argument. These groups are defined in the table below.

Messages	Description
<code>Measurements</code>	<code>+MeasEpoch +MeasExtra +IQCorr +EndOfMeas</code>
<code>GPS</code>	<code>+GPSNav +GPSAlm +GPSIon +GPSUtc</code>
<code>GLO</code>	<code>+GLONav +GLOAlm +GLOTime</code>
<code>GAL</code>	<code>+GALNav +GALAlm +GALIon +GALUtc +GALGstGps</code>
<code>GEO</code>	<code>+GEONav +GEOAlm</code>
<code>CMP</code>	<code>+CMPNav</code>
<code>QZS</code>	<code>+QZSNav</code>
<code>PVTCart</code>	<code>+PVTCartesian +PosCovCartesian +VelCovCartesian +BaseVectorCart</code>
<code>PVTGeod</code>	<code>+PVTGeodetic +PosCovGeodetic +VelCovGeodetic +BaseVectorGeod +PosLocal</code>
<code>PVTEExtra</code>	<code>+DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +GEOCorrections +BaseLine +PVTSupport +EndOfPVT</code>
<code>Attitude</code>	<code>+AttEuler +AttCovEuler +EndOfAtt</code>
<code>Time</code>	<code>+ReceiverTime</code>

Messages (Continued)	Description
Status	+SatVisibility +ChannelStatus +ReceiverStatus +InputLink +OutputLink +IPStatus +NTRIPClientStatus +QualityInd +DiskStatus
LBand	+LBandTrackerStatus +LBAS1DecoderStatus +LBAS1Messages +LBandBeams
UserGroups	+Group1 +Group2 +Group3 +Group4
Rinex	+MeasEpoch +GPSNav +GPSIon +GPSUtc +GLONav +GALNav +GALUtc +GALGstGps +GEONav +CMPNav +QZSNav +PVTGeodetic +ReceiverSetup +Comment
Support	+MeasEpoch +MeasExtra +EndOfMeas +GPSNav +GPSAlm +GPSIon +GPSUtc +GLONav +GLOAlm +GLOTime +GALNav +GALAlm +GALIon +GALUtc +GALGstGps +GEONav +GEOAlm +CMPNav +QZSNav +PVTGeodetic +PosCovGeodetic +BaseVectorGeod +AttEuler +DOP +PVTSupport +EndOfPVT +ChannelStatus +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +Commands +LBandTrackerStatus +LBAS1DecoderStatus +IPStatus +NTRIPClientStatus +QualityInd +LBandBeams +DiskStatus
RawData	+MeasEpoch +MeasExtra +GPSNav +GLONav +GALNav +GEONav +CMPNav +QZSNav +PVTGeodetic +ReceiverSetup +Commands +Comment
PostProcess	+MeasEpoch +MeasExtra +GPSNav +GPSIon +GPSUtc +GLONav +GLOTime +GALNav +GALIon +GALUtc +GALGstGps +GEONav +CMPNav +QZSNav +ReceiverSetup +Commands
GUI	+MeasEpoch +EndOfMeas +EndOfPVT +SatVisibility +ChannelStatus +Commands +PVTGeodetic +PosCovGeodetic +VelCovGeodetic +DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +BaseLine +AttEuler +ReceiverTime +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +Comment +IPStatus +NTRIPClientStatus +QualityInd +DiskStatus

Example

To output the next MeasEpoch block, use:

```
COM1> esoc, COM1, MeasEpoch &ltCR>
$R: esoc, COM1, MeasEpoch
    SBFOnce, COM1, MeasEpoch
COM1>
```

ss0 gso	setSBFOutput getSBFOutput	Stream Stream	Cd	Messages	Interval					
		+ Stream1 ... Stream10 + Res1 + Res2 + Res3 + Res4 all	none DSK1 COM1 COM2 COM3 COM4 USB1 USB2 IP10 ... IP17 NTR1 NTR2 NTR3 IPS1 IPS2 IPS3	none [SBF List] + Measurements + RawNavBits + GPS + GLO + GAL + GEO + CMP + QZS + PVTCart + PVTGeod + PVTExtra + Attitude + Time + Event + DiffCorr + Status + LBand + UserGroups + Rinex + Support + RawData + PostProcess + GUI	off OnChange msec10 msec20 msec40 msec50 msec100 msec200 msec500 sec1 sec2 sec5 sec10 sec15 sec30 sec60 min2 min5 min10 min15 min30 min60					

RxControl: [Communication > Output Settings > SBF Output](#)

Web Interface: [Configuration > Communication > Output Settings > SBF Output](#)

Use this command to output a set of SBF blocks on a given connection at a regular interval.

A *Stream* is defined as a list of messages that should be output with the same interval on one connection descriptor (*Cd*). In other words, one *Stream* is associated with one *Cd* and one *Interval*, and contains a list of SBF blocks defined by the *Messages* argument.

The list of supported SBF blocks [SBF List] is to be found in Section 2.3.13, "SBF List".

Predefined groups of SBF blocks (such as `Measurements` or `RawNavBits`) can be addressed in the *Messages* argument. These groups are defined in the table below.

Messages	Description
Measurements	+MeasEpoch +MeasExtra +IQCorr +ISMR +EndOfMeas
RawNavBits	+GPSRawCA +GPSRawL2C +GPSRawL5 +GLORawCA +GALRawFNAV +GALRawINAV +GEORawL1 +GEORawL5 +CMPRaw +QZSRawL1CA +QZSRawL2C +QZSRawL5
GPS	+GPSNav +GPSAlm +GPSIon +GPSUtc
GLO	+GLONav +GLOAlm +GLOTime
GAL	+GALNav +GALAlm +GALIon +GALUtc +GALGstGps +GALSARRLM
GEO	+GEOMT00 +GEOPRNMmask +GEOfastCorr +GEOIntegrity +GEOfastCorrDegr +GEONav +GEODegrFactors +GEONetworkTime +GEOAlm +GEOIGPMask +GEOLongTermCorr +GEOIonDelay +GEOServiceLevel +GEOClockEphCovMatrix
CMP	+CMPNav
QZS	+QZSNav
PVTCart	+PVTCartesian +PosCovCartesian +VelCovCartesian +BaseVectorCart

Messages (Continued)	Description
PVTGeod	+PVTGeodetic +PosCovGeodetic +VelCovGeodetic +BaseVectorGeod +PosLocal
PVTExtra	+DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +GEOCorrections +BaseLine +PVTSupport +EndOfPVT
Attitude	+AttEuler +AttCovEuler +EndOfAtt
Time	+ReceiverTime +xPPSOffset
Event	+ExtEvent +ExtEventPVTCartesian +ExtEventPVTGeodetic
DiffCorr	+DiffCorrIn +BaseStation +RTCMDatum
Status	+SatVisibility +ChannelStatus +ReceiverStatus +InputLink +OutputLink +IPStatus +NTRIPClientStatus +QualityInd +DiskStatus
LBand	+LBandTrackerStatus +LBAS1DecoderStatus +LBAS1Messages +LBandBeams
UserGroups	+Group1 +Group2 +Group3 +Group4
Rinex	+MeasEpoch +GEORawL1 +GPSNav +GPSIon +GPSUtc +GLONav +GALNav +GALUtc +GALGstGps +GEONav +CMPNav +QZSNav +PVTGeodetic +ReceiverSetup +Comment
Support	+MeasEpoch +MeasExtra +EndOfMeas +GPSRawCA +GPSRawL2C +GPSRawL5 +GLORawCA +GALRawFNAV +GALRawINAV +GEORawL1 +GEORawL5 +CMPRaw +QZSRawL1CA +QZSRawL2C +QZSRawL5 +GPSNav +GPSAIm +GPSIon +GPSUtc +GLONav +GLOAIm +GLOTime +GALNav +GALAIm +GALIon +GALUtc +GALGstGps +GEONav +GEOAIm +CMPNav +QZSNav +PVTGeodetic +PosCovGeodetic +BaseVectorGeod +AttEuler +DOP +PVTSupport +EndOfPVT +ExtEvent +DiffCorrIn +BaseStation +ChannelStatus +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +Commands +LBandTrackerStatus +LBAS1DecoderStatus +IPStatus +NTRIPClientStatus +QualityInd +LBandBeams +DiskStatus
RawData	+MeasEpoch +MeasExtra +GPSRawCA +GPSRawL2C +GPSRawL5 +GLORawCA +GALRawFNAV +GALRawINAV +GEORawL1 +GEORawL5 +CMPRaw +QZSRawL1CA +QZSRawL2C +QZSRawL5 +GPSNav +GLONav +GALNav +GEONav +CMPNav +QZSNav +PVTGeodetic +DiffCorrIn +ReceiverSetup +Commands +Comment
PostProcess	+MeasEpoch +MeasExtra +GEORawL1 +GPSNav +GPSIon +GPSUtc +GLONav +GLOTime +GALNav +GALIon +GALUtc +GALGstGps +GALSARRLM +GEONav +CMPNav +QZSNav +DiffCorrIn +ReceiverSetup +Commands
GUI	+MeasEpoch +EndOfMeas +GEOIGPMask +GEOIonoDelay +EndOfPVT +ExtEvent +DiffCorrIn +SatVisibility +ChannelStatus +Commands +PVTGeodetic +PosCovGeodetic +VelCovGeodetic +DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +BaseLine +AttEuler +ReceiverTime +BaseStation +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +Comment +IPStatus +NTRIPClientStatus +QualityInd +DiskStatus

The *Interval* argument defines the rate at which the SBF blocks specified in the *Messages* argument are output. If set to `off`, the SBF blocks are disabled. If set to `OnChange`, the SBF blocks are output at their natural renewal rate. Please refer to the "Output Rate" section of the SBF Reference Guide for further details. If a specific interval is specified (e.g. `sec1` corresponds to an interval of 1 second), the SBF blocks are decimated from their renewal rate to the specified interval. Some blocks can only be output at their renewal rate (e.g. the `GPSNav` block). For these blocks, the receiver ignores any decimation interval and always assumes `OnChange`. The list of those blocks can be found in the SBF Reference Guide.

Please make sure that the connection specified by *Cd* is configured to allow SBF output (this is the default for all connections). See the `setDataInOut` command.

`Res1` to `Res4` are reserved values of *Stream*. These streams are not saved in the configuration files and, as a consequence, they will always be reset at boot time. For most users, it is not recommended to use these streams.

Examples

To output the `MeasEpoch` block at 1Hz and the `PVTCartesian` block at 10Hz on COM1, use the following sequence:

```
COM1> sso, Stream1, COM1, MeasEpoch, sec1 <CR>
$R: sso, Stream1, COM1, MeasEpoch, sec1
    SBFOutput, Stream1, COM1, MeasEpoch, sec1
COM1> sso, Stream2, COM1, PVTCartesian, msec100 <CR>
$R: sso, Stream2, COM1, PVTCartesian, msec100
    SBFOutput, Stream2, COM1, PVTCartesian, msec100
COM1>
```

To get the list of SBF blocks currently output, use:

```
COM1> gso <CR>
$R: gso
    SBFOutput, Stream1, COM1, MeasEpoch, sec1
    SBFOutput, Stream2, COM1, PVTCartesian, msec100
    SBFOutput, Stream3, none, none, off
    SBFOutput, Stream4, none, none, off
    SBFOutput, Stream5, none, none, off
    SBFOutput, Stream6, none, none, off
    SBFOutput, Stream7, none, none, off
    SBFOutput, Stream8, none, none, off
    SBFOutput, Stream9, none, none, off
    SBFOutput, Stream10, none, none, off
    SBFOutput, Res1, none, none, off
    SBFOutput, Res2, none, none, off
    SBFOutput, Res3, none, none, off
    SBFOutput, Res4, none, none, off
COM1>
```


2.3.8 RTCM v2.x Commands

sr2c gr2c	setRTCMv2Compatibility getRTCMv2Compatibility	PRCType	GLOToD							
		Standard GroupDelay	Tk Tb							

RxControl: [Communication > Input Settings > Differential Corrections > RTCMv2](#)

Web Interface: [Configuration > Communication > Input Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire the compatibility of the RTCM 2.x input correction stream. This command applies to rover receivers only and should be used in case the available base station correction stream is not fully compatible with the latest version of the RTCM 2.x standard.

The argument *PRCType* is used to handle a difference in the interpretation of DGPS corrections between the version 2.0 of the RTCM standard and later versions. If the base station is sending RTCM Message Type 1 based on version 2.0, the value *GroupDelay* must be selected to have a correct usage of incoming corrections.

The argument *GLOToD* specifies how to interpret the time-of-day field in the differential GLONASS correction message (MT31). Select *Tb* to be compatible with RTCM version up to 2.2, and select *Tk* to be compatible with RTCM 2.3 and later.

Example

To make to rover receiver compatible with a base station sending RTCM 2.2 corrections, use:

```
COM1> sr2c, , Tb <CR>
$R: sr2c, , Tb
      RTCMv2Compatibility, Standard, Tb
COM1>
```


sr2f gr2f	setRTCMv2Formatting getRTCMv2Formatting	ReferenceID	GLOToD							
		0 ... 1023	Tk Tb							

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv2](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire the reference station ID assigned to the receiver when operating in base station mode. The reference station ID is transmitted in the first word of each outgoing RTCM v2.x message.

The argument *GLOToD* specifies how to encode the time-of-day field in the differential GLONASS correction message (MT31). Select **Tb** to be compatible with RTCM version up to 2.2, and select **Tk** to be compatible with RTCM 2.3 and later.

Examples

```
COM1> sr2f, 345 <CR>
$R: sr2f, 345
    RTCMv2Formatting, 345, Tk
COM1>
```

```
COM1> gr2f <CR>
$R: gr2f
    RTCMv2Formatting, 345, Tk
COM1>
```

sr2i gr2i	setRTCMv2Interval getRTCMv2Interval	Message Message	ZCount							
		+RTCM1 +RTCM3 +RTCM9 +RTCM16 +RTCM17 +RTCM22 +RTCM23 24 +RTCM31 +RTCM32 all	1 ... 2 ... 1000							

RxControl: [Communication > Output Settings > Differential Corrections > RTCMv2](#)

Web Interface: [Configuration > Communication > Output Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire at which interval the RTCM v2.x messages specified in the *Message* argument should be generated. The related **setRTCMv2IntervalObs** command must be used to specify the interval of some RTK-related messages such as messages 18 and 19.

The interval for every message is given in the *ZCount* argument, in units of 0.6 seconds. For example, to generate a message every 6 seconds, *ZCount* should be set to 10.

For the ephemerides message (RTCM17), the ephemerides are sent out one satellite at a time, at a rate specified by this command. For instance, if *ZCount* is set to 1 and there are 12 ephemerides to send out, it takes $0.6 \times 12 = 7.2$ seconds to send the whole ephemerides set.

The intervals specified with this command are not connection-specific: all the connections which output a given RTCM v2.x message will output it with the same interval.

Note that this command only defines the interval of RTCM messages. To make the receiver actually output these messages, use the **setRTCMv2Output** and **setDataInOut** commands.

See the **setRTCMv2Usage** command for a short description of all supported RTCM v2.x message types.

Examples

```
COM1> sr2i, RTCM22, 15 <CR>
```

```
$R: sr2i, RTCM22, 15
```

```
RTCMv2Interval, RTCM22, 15
```

```
COM1>
```

```
COM1> gr2i <CR>
```

```
$R: gr2i
```

```
RTCMv2Interval, RTCM1, 2
```

```
RTCMv2Interval, RTCM3, 2
```

```
RTCMv2Interval, RTCM16, 2
```

```
RTCMv2Interval, RTCM22, 15
```

```
RTCMv2Interval, RTCM23|24, 2
```

```
COM1>
```

sr2b gr2b	setRTCMv2IntervalObs getRTCMv2IntervalObs	Message Message	Interval						
		+ RTCM18 19 + RTCM20 21 all	1 ... 600 sec						

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv2](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire at which interval the RTCM v2.x messages specified in the *Message* argument should be generated. The related **setRTCMv2Interval** command must be used to specify the interval of other supported RTCM v2.x messages.

The intervals specified with this command are not connection-specific: all the connections which output a given RTCM v2.x message will output it with the same interval.

Note that this command only defines the interval of RTCM messages. To make the receiver actually output these messages, use the **setRTCMv2Output** and **setDataInOut** commands.

Examples

```
COM1> sr2b, RTCM20|21, 2 <CR>
$R: sr2b, RTCM20|21, 2
    RTCMv2IntervalObs, RTCM20|21, 2
COM1>
```

```
COM1> gr2b <CR>
$R: gr2b
    RTCMv2IntervalObs, RTCM18|19, 1
    RTCMv2IntervalObs, RTCM20|21, 2
COM1>
```

sr2m	setRTCMv2Message16	Message (90)								
gr2m	getRTCMv2Message16									
		Unknown								

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv2](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire the string that will be transmitted in the RTCM v2.x message 16. The argument *Message* can contain up to 90 characters.

Note that this command only defines the content of message 16. To make the receiver actually output this message, use the **setRTCMv2Output** and **setDataInOut** commands.

Example

To send the string "Hello" in message 16 over COM2 at the default interval, use the following sequence:

```
COM1> sr2m, Hello <CR>
$R: sr2m, Hello
      RTCMv2Message16, "Hello"
COM1> sr2o, COM2, RTCM16 <CR>
$R: sr2o, COM2, RTCM16
      RTCMv2Output, COM2, RTCM16
COM1> sdio, COM2, , RTCMv2 <CR>
$R: sdio, COM2, , RTCMv2
      DataInOut, COM2, auto, RTCMv2
COM1>
```

sr2o gr2o	setRTCMv2Output getRTCMv2Output	Cd Cd	Messages							
		+ COM1	none							
		+ COM2	+ RTCM1							
		+ COM3	+ RTCM3							
		+ COM4	+ RTCM9							
		+ USB1	+ RTCM16							
		+ USB2	+ RTCM18 19							
		+ IP10 ... IP17	+ RTCM20 21							
		+ NTR1	+ RTCM22							
		+ NTR2	+ RTCM23 24							
		+ NTR3	+ RTCM31							
		+ IPS1	+ RTCM32							
		+ IPS2	+ RTCM17							
		+ IPS3	+ DGPS							
		all	+ RTK							
			all							

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv2](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv2](#)

Use these commands to define/inquire which RTCM v2.x messages are enabled for output on a given connection descriptor (*Cd*). The *Messages* argument specifies the RTCM message types to be enabled. Some pairs of messages are always enabled together, such as messages 18 and 19. DGPS is an alias for "RTCM1+RTCM3" and RTK is an alias for "RTCM3+RTCM18|19+RTCM22".

See the **setRTCMv2Usage** command for a short description of all supported RTCM v2.x message types.

Please make sure that the connection specified by *Cd* is configured to allow RTCMv2 output, which can be done with the **setDataInOut** command. The interval at which each message is output is to be specified with the **setRTCMv2Interval** or the **setRTCMv2IntervalObs** command.

Example

To enable RTCM v2.x messages 3, 18, 19 and 22 on COM2, use the following sequence:

```
COM1> sr2o, COM2, RTCM3+RTCM18|19+RTCM22 <CR>
$R: sr2o, COM2, RTCM3+RTCM18|19+RTCM22
    RTCMv2Output, COM2, RTCM3+RTCM18|19+RTCM22
COM1> sdio, COM2, , RTCMv2 <CR>
$R: sdio, COM2, , RTCMv2
    DataInOut, COM2, auto, RTCMv2
COM1>
```

sr2u gr2u	setRTCMv2Usage getRTCMv2Usage	MsgUsage								
		none + RTCM1 + RTCM3 + RTCM9 + RTCM15 + RTCM18 19 + RTCM20 21 + RTCM22 + RTCM23 24 + RTCM31 + RTCM32 + RTCM17 + RTCM59 all								

RxControl: [Communication](#) > [Input Settings](#) > [Differential Corrections](#) > [RTCMv2](#)

Web Interface: [Configuration](#) > [Communication](#) > [Input Settings](#) > [Differential Corrections](#) > [RTCMv2](#)

Use this command to restrict the list of incoming RTCM v2.x messages that the receiver is allowed to use in its differential PVT computation.

MsgUsage	Description
RTCM1	Differential GPS Corrections
RTCM3	GPS Reference Station Parameters
RTCM9	GPS Partial Correction Set
RTCM15	Ionospheric Delay
RTCM18 19	RTK Uncorrected Carrier Phases (RTCM18) and Pseudoranges (RTCM19)
RTCM20 21	RTK Carrier Phase Corrections (RTCM20) and High-Accuracy Pseudorange Corrections (RTCM21)
RTCM22	Extended Reference Station Parameters
RTCM23 24	Antenne Type Definition Record (RTCM23) and Antenna Reference Point (ARP) (RTCM24)
RTCM31	Differential GLONASS Corrections
RTCM32	GLONASS Reference Station Parameters
RTCM17	GPS Ephemerides Message
RTCM59	Proprietary Message (interpreted as FKP message)

Example

To only accept RTCM1 and RTCM3 corrections from the base station 1011, use the following sequence:

```
COM1> sr2u, RTCM1+RTCM3 <CR>
$R: sr2u, RTCM1+RTCM3
    RTCMv2Usage, RTCM1+RTCM3
COM1> sdcu, , , manual, 1011 <CR>
$R: sdcu, , , manual, 1011
    DiffCorrUsage, LowLatency, 3600.0, manual, 1011, 20, 20000000
COM1>
```

2.3.9 RTCM v3.x Commands

sr3t gr3t	setRTCMv3CRSTransfo getRTCMv3CRSTransfo	Mode	TargetName (32)							
		auto manual								

RxControl: Communication > Input Settings > Differential Corrections > RTCMv3

Web Interface: Configuration > Communication > Input Settings > Differential Corrections > RTCMv3

Use this command to specify how to apply the coordinate reference system (CRS) transformation parameters contained in RTCM v3.x message types 1021 to 1023.

In `auto` mode (the default), the receiver decodes and applies the coordinate transformation parameters from message types 1021-1023. If your RTK provider sends transformation parameters for more than one target CRS, the receiver selects the first transformation parameters it receives.

In `manual` mode, you can force the receiver to only apply the transformation to the target CRS specified with the second argument. The *TargetName* argument must exactly match the name used by the RTK provider. The available target datum names can be found in the `RTCMDatum` SBF block.

Example

To force using the target CRS identified as "4258" by the RTK network, use:

```
COM1> sr3t, manual, "4258" <CR>
$R: sr3t, manual, "4258"
      RTCMv3CRSTransfo, manual, "4258"
COM1>
```


sr3d	setRTCMv3Delay	Delay								
gr3d	getRTCMv3Delay									
		0 ... 600 sec								

RxControl: Communication > Output Settings > Differential Corrections > RTCMv3

Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv3

Use this command to instruct the receiver to generate and output RTCM v3.x messages with a certain delay.

It is possible to impose a global delay to all RTCM v3.x messages by setting the *Delay* to a non-zero value. This can be used in situations where multiple base stations must be configured to transmit their corrections in a time-multiplexed way. For example, base station A would compute and transmit its corrections at every 10-second epoch (in the GPS time scale), and base station B would compute and transmit its corrections 5 seconds after the 10-second epochs. In that case, receiver B would be configured with the *Delay* argument set to 5.

See also the **setRTCMv3Interval** command to configure the message interval.

Example

To generate the RTCM1001 message with an interval of 10 seconds and a time shift of 2 seconds, use:

```
COM1> sr3i, RTCM1001|2, 10 <CR>
$R: sr3i, RTCM1001|2, 10
    RTCMv3Interval, RTCM1001|2, 10
COM1> sr3d, 2 <CR>
$R: sr3d, 2
    RTCMv3Delay, 2
COM1>
```

sr3f gr3f	setRTCMv3Formatting getRTCMv3Formatting	ReferenceID	MSMSignals	GLOL2						
		0 ... 4095	+GPSL1CA +GPSL2PY +GPSL2C +GPSL5 +GLOL1CA +GLOL2P +GLOL2CA +GLOL3 +GALL1BC +GALE5a +GALE5b +GALE5 +CMPL1 +CMPE5b +QZSL1CA +QZSL2C +QZSL5 all	L2CA L2P						

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv3](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv3](#)

Use these commands to configure the RTCM v3.x message contents when operating in base station mode.

The *ReferenceID* argument specifies the reference station ID transmitted in the header of each outgoing RTCM v3.x message.

The *MSMSignals* argument specifies the signal types to be encoded in MSM messages. For an observable to be actually encoded in MSM, the corresponding signal type must be enabled with this command, the signal must be enabled for tracking (see the **setSignalTracking** command), and a suitable MSM message must be enabled with the **setRTCMv3Output** command.

The *GLOL2* argument applies to message types 1011 and 1012 (GLONASS L1 and L2 observables). It specifies which of the L2P or the L2CA observables must be encoded in RTCM1011 and RTCM1012.

Example

```
COM1> sr3f, 345 <CR>
$R: sr3f, 345
    RTCMv3Formatting, 345, GPSL1CA+GPSL2PY+GLOL1CA+GLOL2CA+GALL1BC+
    GALE5a+CMPL1+CMPE5b+QZSL1CA+QZSL2C, L2CA
COM1>
```

sr3i gr3i	setRTCMv3Interval getRTCMv3Interval	Message Message	Interval							
		+RTCM1001 2 +RTCM1003 4 +RTCM1005 6 +RTCM1007 8 +RTCM1009 10 +RTCM1011 12 +RTCM1013 +RTCM1019 +RTCM1020 +RTCM1029 +RTCM1033 +RTCM1044 +RTCM1045 +MSM1 ... MSM7 all	0.1 ... 1.0 ... 600.0 sec							

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv3](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv3](#)

Use these commands to define/inquire at which interval RTCM v3.x messages should be generated.

The intervals specified with this command are not connection-specific: all the connections which output a given RTCM v3.x message will output it with the same interval.

Using `MSMi` for the *Message* argument sets the interval of all Multiple Signal Messages of type *i*. See the `setRTCMv3Output` command for a short description of all supported RTCM v3.x message types.

For the ephemerides messages (e.g. RTCM1019), the ephemerides are sent out one satellite at a time, at a rate specified by this command. For instance, if *Interval* is set to 1 and there are 12 GPS ephemerides to send out, it takes 12 seconds to send the whole GPS ephemerides set.

By default, RTCM v3.x messages are generated at integer multiples of the specified interval in the GPS time scale. The command `setRTCMv3Delay` can be used to introduce a time offset.

Note that this command only defines the interval of RTCM messages. To make the receiver actually output these messages, use the `setRTCMv3Output` and `setDataInOut` commands.

Example

```
COM1> sr3i, RTCM1001|2, 2 <CR>
$R: sr3i, RTCM1001|2, 2
    RTCMv3Interval, RTCM1001|2, 2
COM1>
```

sr3m	setRTCMv3Message1029	Message (120)								
gr3m	getRTCMv3Message1029	Unknown								

[RxControl: Communication > Output Settings > Differential Corrections > RTCMv3](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > RTCMv3](#)

Use these commands to define/inquire the string that will be transmitted in the RTCM v3.x message 1029. The argument *Message* can contain up to 120 characters.

Note that this command only defines the content of message 1029. To make the receiver actually output this message, use the **setRTCMv3Output** and **setDataInOut** commands.

Example

To send the string "Hello" in message 1029 over COM2 at the default interval, use the following sequence:

```
COM1> sr3m, Hello <CR>
$R: sr3m, Hello
      RTCMv3Message1029, "Hello"
COM1> sr3o, COM2, RTCM1029 <CR>
$R: sr3o, COM2, RTCM1029
      RTCMv3Output, COM2, RTCM1029
COM1> sdio, COM2, , RTCMv3 <CR>
$R: sdio, COM2, , RTCMv3
      DataInOut, COM2, auto, RTCMv3
COM1>
```

sr3o gr3o	setRTCMv3Output getRTCMv3Output	Cd Cd	Messages							
		+ COM1	none							
		+ COM2	+ RTCM1001							
		+ COM3	+ RTCM1002							
		+ COM4	+ RTCM1003							
		+ USB1	+ RTCM1004							
		+ USB2	+ RTCM1005							
		+ IP10 ... IP17	+ RTCM1006							
		+ NTR1	+ RTCM1007							
		+ NTR2	+ RTCM1008							
		+ NTR3	+ RTCM1009							
		+ IPS1	+ RTCM1010							
		+ IPS2	+ RTCM1011							
		+ IPS3	+ RTCM1012							
		all	+ RTCM1013							
			+ RTCM1019							
			+ RTCM1020							
			+ RTCM1029							
			+ RTCM1033							
			+ RTCM1044							
			+ RTCM1045							
			+ RTCM1071 ... RTCM1077							
			+ RTCM1081 ... RTCM1087							
			+ RTCM1091 ... RTCM1097							
			+ RTCM1111 ... RTCM1117							
			+ RTCM1121 ... RTCM1127							
			+ MSM1							
			+ MSM2							
			+ MSM3							
			+ MSM4							
			+ MSM5							
			+ MSM6							
			+ MSM7							
			all							

RxControl: [Communication > Output Settings > Differential Corrections > RTCMv3](#)

Web Interface: [Configuration > Communication > Output Settings > Differential Corrections > RTCMv3](#)

Use these commands to define/inquire which RTCM v3.x messages are enabled for output on a given connection descriptor (*Cd*). The *Messages* argument specifies the RTCM message types to be enabled.

A short description of the supported RTCM v3.x message types is shown below. *MSMi* enables the Multiple Signal Message - Type i(MSMi) from all constellations.

Please make sure that the connection specified by *Cd* is configured to allow RTCMv3 output, which can be done with the **setDataInOut** command. The interval at which each message is output is to be specified with the **setRTCMv3Interval** command.

Messages	Description
RTCM1001	L1-Only GPS RTK Observables
RTCM1002	Extended L1-Only GPS RTK Observables
RTCM1003	L1&L2 GPS RTK Observables
RTCM1004	Extended L1&L2 GPS RTK Observables
RTCM1005	Stationary RTK Reference Station ARP
RTCM1006	Stationary RTK Reference Station ARP with Antenna Height
RTCM1007	Antenna Descriptor
RTCM1008	Antenna Descriptor and Serial Number

Messages (Continued)	Description
RTCM1009	L1-Only GLONASS RTK Observables
RTCM1010	Extended L1-Only GLONASS RTK Observables
RTCM1011	L1&L2 GLONASS RTK Observables
RTCM1012	Extended L1&L2 GLONASS RTK Observables
RTCM1013	System Parameters
RTCM1019	GPS Satellite Ephemeris Data
RTCM1020	Glomass Satellite Ephemeris Data
RTCM1029	Unicode Text String
RTCM1033	Receiver and Antenna Descriptors
RTCM1044	QZSS Satellite Ephemeris Data
RTCM1045	Galileo F/NAV Satellite Ephemeris Data
RTCM1071	GPS MSM1, Compact GPS Pseudoranges
RTCM1072	GPS MSM2, Compact GPS PhaseRanges
RTCM1073	GPS MSM3, Compact GPS Pseudoranges and PhaseRanges
RTCM1074	GPS MSM4, Full GPS Pseudoranges and PhaseRanges plus CNR
RTCM1075	GPS MSM5, Full GPS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1076	GPS MSM6, Full GPS Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1077	GPS MSM7, Full GPS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1081	GLONASS MSM1, Compact GLONASS Pseudoranges
RTCM1082	GLONASS MSM2, Compact GLONASS PhaseRanges
RTCM1083	GLONASS MSM3, Compact GLONASS Pseudoranges and PhaseRanges
RTCM1084	GLONASS MSM4, Full GLONASS Pseudoranges and PhaseRanges plus CNR
RTCM1085	GLONASS MSM5, Full GLONASS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1086	GLONASS MSM6, Full GLONASS Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1087	GLONASS MSM7, Full GLONASS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1091	GALILEO MSM1, Compact GALILEO Pseudoranges
RTCM1092	GALILEO MSM2, Compact GALILEO PhaseRanges
RTCM1093	GALILEO MSM3, Compact GALILEO Pseudoranges and PhaseRanges
RTCM1094	GALILEO MSM4, Full GALILEO Pseudoranges and PhaseRanges plus CNR
RTCM1095	GALILEO MSM5, Full GALILEO Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1096	GALILEO MSM6, Full GALILEO Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1097	GALILEO MSM7, Full GALILEO Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1111	QZSS MSM1, Compact QZSS Pseudoranges
RTCM1112	QZSS MSM2, Compact QZSS PhaseRanges
RTCM1113	QZSS MSM3, Compact QZSS Pseudoranges and PhaseRanges

Messages (Continued)	Description
RTCM1114	QZSS MSM4, Full QZSS Pseudoranges and PhaseRanges plus CNR
RTCM1115	QZSS MSM5, Full QZSS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1116	QZSS MSM6, Full QZSS Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1117	QZSS MSM7, Full QZSS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1121	BEIDOU MSM1, Compact BEIDOU Pseudoranges
RTCM1122	BEIDOU MSM2, Compact BEIDOU PhaseRanges
RTCM1123	BEIDOU MSM3, Compact BEIDOU Pseudoranges and PhaseRanges
RTCM1124	BEIDOU MSM4, Full BEIDOU Pseudoranges and PhaseRanges plus CNR
RTCM1125	BEIDOU MSM5, Full BEIDOU Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1126	BEIDOU MSM6, Full BEIDOU Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1127	BEIDOU MSM7, Full BEIDOU Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)

Example

To enable RTCM v3.x messages 1001, 1002, 1005 and 1006 on COM2, use the following sequence:

```
COM1> sr3o, COM2, RTCM1001+RTCM1002+RTCM1005+RTCM1006 <CR>
$R: sr3o, COM2, RTCM1001+RTCM1002+RTCM1005+RTCM1006
    RTCMv3Output, COM2, RTCM1001+RTCM1002+RTCM1005+RTCM1006
COM1> sdio, COM2, , RTCMv3 <CR>
$R: sdio, COM2, , RTCMv3
    DataInOut, COM2, auto, RTCMv3
COM1>
```


sr3u gr3u	setRTCMv3Usage getRTCMv3Usage	MsgUsage								
		none + RTCM1001 ... RTCM1013 + RTCM1015 + RTCM1016 + RTCM1017 + RTCM1019 + RTCM1021 ... RTCM1027 + RTCM1033 + RTCM1037 + RTCM1038 + RTCM1039 + RTCM1071 ... RTCM1077 + RTCM1081 ... RTCM1087 + RTCM1121 ... RTCM1127 + RTCM1029 all								

RxControl: Communication > Input Settings > Differential Corrections > RTCMv3

Web Interface: Configuration > Communication > Input Settings > Differential Corrections > RTCMv3

Use this command to restrict the list of incoming RTCM v3.x messages that the receiver is allowed to use in its differential PVT computation.

A short description of the supported RTCM v3.x message types is shown below:

MsgUsage	Description
RTCM1001	L1-Only GPS RTK Observables
RTCM1002	Extended L1-Only GPS RTK Observables
RTCM1003	L1&L2 GPS RTK Observables
RTCM1004	Extended L1&L2 GPS RTK Observables
RTCM1005	Stationary RTK Reference Station ARP
RTCM1006	Stationary RTK Reference Station ARP with Antenna Height
RTCM1007	Antenna Descriptor
RTCM1008	Antenna Descriptor and Serial Number
RTCM1009	L1-Only GLONASS RTK Observables
RTCM1010	Extended L1-Only GLONASS RTK Observables
RTCM1011	L1&L2 GLONASS RTK Observables
RTCM1012	Extended L1&L2 GLONASS RTK Observables
RTCM1013	System Parameters
RTCM1015	Network RTK (MAC), GPS Ionospheric Correction Differences
RTCM1016	Network RTK (MAC), GPS Geometric Correction Differences
RTCM1017	Network RTK (MAC), GPS Combined Geometric and Ionospheric Correction Differences
RTCM1019	GPS Satellite Ephemeris Data
RTCM1021	Helmert-Abridged Molodenski Transformation Parameters
RTCM1022	Molodenski-Badekas Transformation Parameters
RTCM1023	Residuals, Ellipsoidal Grid Representation
RTCM1024	Residuals, Plane Grid Representation
RTCM1025	Projection Parameters, Projection Types other than Lambert Conic Conformal (2 SP) and Oblique Mercator
RTCM1026	Projection Parameters, Projection Type LCC2SP (Lambert Conic Conformal (2 SP))

MsgUsage (Continued)	Description
RTCM1027	Projection Parameters, Projection Type OM (Oblique Mercator)
RTCM1033	Receiver and Antenna Descriptors
RTCM1037	Network RTK (MAC), GLONASS Ionospheric Correction Differences
RTCM1038	Network RTK (MAC), GLONASS Geometric Correction Differences
RTCM1039	Network RTK (MAC), GLONASS Combined Geometric and Ionospheric Correction Differences
RTCM1071	GPS MSM1, Compact GPS Pseudoranges
RTCM1072	GPS MSM2, Compact GPS PhaseRanges
RTCM1073	GPS MSM3, Compact GPS Pseudoranges and PhaseRanges
RTCM1074	GPS MSM4, Full GPS Pseudoranges and PhaseRanges plus CNR
RTCM1075	GPS MSM5, Full GPS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1076	GPS MSM6, Full GPS Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1077	GPS MSM7, Full GPS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1081	GLONASS MSM1, Compact GLONASS Pseudoranges
RTCM1082	GLONASS MSM2, Compact GLONASS PhaseRanges
RTCM1083	GLONASS MSM3, Compact GLONASS Pseudoranges and PhaseRanges
RTCM1084	GLONASS MSM4, Full GLONASS Pseudoranges and PhaseRanges plus CNR
RTCM1085	GLONASS MSM5, Full GLONASS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1086	GLONASS MSM6, Full GLONASS Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1087	GLONASS MSM7, Full GLONASS Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1121	BEIDOU MSM1, Compact BEIDOU Pseudoranges
RTCM1122	BEIDOU MSM2, Compact BEIDOU PhaseRanges
RTCM1123	BEIDOU MSM3, Compact BEIDOU Pseudoranges and PhaseRanges
RTCM1124	BEIDOU MSM4, Full BEIDOU Pseudoranges and PhaseRanges plus CNR
RTCM1125	BEIDOU MSM5, Full BEIDOU Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1126	BEIDOU MSM6, Full BEIDOU Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1127	BEIDOU MSM7, Full BEIDOU Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1029	Unicode Text String

Example

To only accept RTCM1001 and RTCM1002 corrections from the base station 1011, use the following sequence:

```
COM1> sr3u, RTCM1001+RTCM1002 <CR>
$R: sr3u, RTCM1001+RTCM1002
    RTCMv3Usage, RTCM1001+RTCM1002
COM1> sdcu, , , manual, 1011 <CR>
$R: sdcu, , , manual, 1011
    DiffCorrUsage, LowLatency, 3600.0, manual, 1011, 20, 20000000
COM1>
```

2.3.10 CMR v2.0 Commands

sc2f	setCMRv2Formatting	ReferenceID								
gc2f	getCMRv2Formatting									
		0...31								

RxControl: Communication > Output Settings > Differential Corrections > CMRv2

Web Interface: Configuration > Communication > Output Settings > Differential Corrections > CMRv2

Use these commands to define/inquire the reference station ID assigned to the receiver when operating in base station mode. The reference station ID is transmitted in the header of each outgoing CMR v2.0 message.

Examples

```
COM1> sc2f, 12 <CR>
$R: sc2f, 12
    CMRv2Formatting, 12
COM1>
```

```
COM1> gc2f <CR>
$R: gc2f
    CMRv2Formatting, 12
COM1>
```

sc2i gc2i	setCMRv2Interval getCMRv2Interval	Message Message	Interval							
		+CMR0 +CMR1 +CMR2 +CMR3 all	0.1 ... 1.0 ... 600.0 sec							

RxControl: Communication > Output Settings > Differential Corrections > CMRv2

Web Interface: Configuration > Communication > Output Settings > Differential Corrections > CMRv2

Use these commands to define/inquire at which interval CMR v2.0 messages should be generated.

The intervals specified with this command are not connection-specific: all the connections which output a given CMR v2.0 message will output it with the same interval.

Note that this command only defines the interval of CMR messages. To make the receiver actually output these messages, use the **setCMRv2Output** and **setDataInOut** commands.

See the **setCMRv2Usage** command for a short description of the supported CMR v2.0 messages.

Examples

```
COM1> sc2i, CMR0, 2 <CR>
```

```
$R: sc2i, CMR0, 2
```

```
CMRv2Interval, CMR0, 2
```

```
COM1>
```

```
COM1> gc2i <CR>
```

```
$R: gc2i CMRv2Interval, CMR0, 2
```

```
CMRv2Interval, CMR1, 1 CMRv2Interval, CMR2, 1
```

```
COM1>
```

sc2m	setCMRv2Message2	ShortID (8)	LongID (50)	COGO (16)						
gc2m	getCMRv2Message2									
		Unknown	Unknown	Unknown						

[RxControl: Communication > Output Settings > Differential Corrections > CMRv2](#)

[Web Interface: Configuration > Communication > Output Settings > Differential Corrections > CMRv2](#)

Use these commands to define/inquire the strings that will be transmitted in the CMR v2.0 message 2.

The argument *ShortID* is the short station ID. It can contain up to 8 characters in compliance with the CMR standard. If less than 8 characters are defined, the string will be right justified and padded with spaces.

The argument *LongID* is the long station ID. It can contain up to 50 characters in compliance with the CMR standard. If less than 50 characters are defined, the string will be right justified and padded with spaces. Some CMR implementations use the character "@" in the long name. As this character is not allowed in a command argument, the character "%" should be used instead. The receiver will automatically replace all occurrences of "%" in *LongID* with "@" when CMR2 message is output.

The argument *COGO* is the COGO code. It can contain up to 16 characters in compliance with the CMR standard. If less than 16 characters are defined, the string will be right justified and padded with spaces.

Note that this command only defines the contents of message 2. To make the receiver actually output this message, use the **setCMRv2Output** and **setDataInOut** commands.

Example

To send the string "Hello" as short station ID and send CMR2 messages through COM2, use the following sequence:

```
COM1> sc2m, Hello <CR>
$R: sc2m, Hello
      CMRv2Message2, "Hello", "Unknown", "Unknown"
COM1> sc2o, COM2, CMR2 <CR>
$R: sc2o, COM2, CMR2
      CMRv2Output, COM2, CMR2
COM1> sdio, COM2, , CMRv2 <CR>
$R: sdio, COM2, , CMRv2
      DataInOut, COM2, auto, CMRv2
COM1>
```

sc2o gc2o	setCMRv2Output getCMRv2Output	Cd Cd	Messages							
		+COM1 +COM2 +COM3 +COM4 +USB1 +USB2 +IP10 ... IP17 +NTR1 +NTR2 +NTR3 +IPS1 +IPS2 +IPS3 all	none +CMR0 +CMR1 +CMR2 +CMR3 all							

RxControl: Communication > Output Settings > Differential Corrections > CMRv2

Web Interface: Configuration > Communication > Output Settings > Differential Corrections > CMRv2

Use these commands to define/inquire which CMR v2.0 messages are enabled for output on a given connection descriptor (*Cd*). The *Messages* argument specifies the CMR message types to be enabled. See the **setCMRv2Usage** command for a short description of the supported CMR v2.0 messages.

Please make sure that the connection specified by *Cd* is configured to allow CMRv2 output, which can be done with the **setDataInOut** command. The interval at which each message is output is to be specified with the **setCMRv2Interval** command.

Example

To enable CMR v2.0 message 0 on COM2, use the following sequence:

```
COM1> sc2o, COM2, CMR0 <CR>
$R: sc2o, COM2, CMR0
    CMRv2Output, COM2, CMR0
COM1> sdio, COM2, , CMRv2 <CR>
$R: sdio, COM2, , CMRv2
    DataInOut, COM2, auto, CMRv2
COM1>
```


sc2u gc2u	setCMRv2Usage getCMRv2Usage	MsgUsage								
		none +CMR0 +CMR1 +CMR2 +CMR3 +CMR0p +CMR0w all								

RxControl: Communication > Input Settings > Differential Corrections > CMRv2

Web Interface: Configuration > Communication > Input Settings > Differential Corrections > CMRv2

Use this command to restrict the list of incoming CMR v2.0 messages that the receiver is allowed to use in its differential PVT computation. CMR0p and CMR0w refer to the CMR+ and CMR-W variants respectively.

MsgUsage	Description
CMR0	Observables
CMR1	Reference Station Coordinates
CMR2	Reference Station Description
CMR3	GLONASS Observables

Example

To only accept CMR0 from the base station 12, use the following sequence:

```
COM1> sc2u, CMR0 <CR>
$R: sc2u, CMR0
      CMRv2Usage, CMR0
COM1> sdcu, , , manual, 12 <CR>
$R: sdcu, , , manual, 12
      DiffCorrUsage, LowLatency, 3600.0, manual, 12, 20, 20000000
COM1>
```

2.3.11 Logging Commands

sdfa gdfa	setDiskFullAction getDiskFullAction	Disk Disk	Action							
		+ DSK1 all	DeleteOldest StopLogging							

RxControl: Logging > Internal Logging Settings

Web Interface: Logging > Internal Logging Settings

Use these commands to define/inquire what the receiver should do when the disk identified by *Disk* is full, or when an auto-incremented file name already exists on that disk (see command **setFileNaming** for a description of the incremental file naming mode).

The currently supported actions are as follows:

Action	Description
DeleteOldest	The oldest file on the disk is automatically removed, unless the oldest file is also the current logging file. In that latter case, the logging stops. In incremental file naming mode, if the auto-incremented file name already exists, the existing file is overwritten.
StopLogging	The logging stops. In incremental file naming mode, if the auto-incremented file name already exists, the logging stops.

Examples

```
COM1> sdfa, DSK1, StopLogging <CR>
$R: sdfa, DSK1, StopLogging
    DiskFullAction, DSK1, StopLogging
COM1>
```

```
COM1> gdfa <CR>
$R: gdfa
    DiskFullAction, DSK1, StopLogging
COM1>
```

ldi	IstDiskInfo	Disk	Directory (60)							
		DSK1 all								

Use this command to retrieve information about one of the internal disks of the receiver. The reply to this command contains the disk size and free space in bytes and the list of all recorded files and directories.

The contents of directories is not shown by default. To list the contents of a directory, use the second argument to specify the directory name.

Example

```
COM1> ldi, DSK1 <CR>
$R; ldi, dsk1
---->
$-- BLOCK 1 / 0
<xml version="1.0" encoding="ISO-8859-1" ?>
<DiskInfo version="0.1">
  <Disk name="DSK1" total="2030927872" free="2030764032" >
    <File name="log.sbf" size="16384" locked="yes" />
    <File name="leuv2050.07_" size="35196" locked="no" />
  </Disk>
</DiskInfo>
COM1>
```

sfn gfn	setFileNaming getFileNaming	Disk Disk	NamingType	FileName (8)						
		+ DSK1 all	FileName Incremental IGS15M IGS1H IGS6H IGS24H	log						

RxControl: [Logging > Internal Logging Settings](#)
Web Interface: [Logging > Internal Logging Settings](#)

Use these commands to define/inquire the file naming convention applied to name the internal SBF, NMEA or user-message log files. This command does not apply to internal RINEX logging, which is configured with the **setRINEXLogging** command.

If *NamingType* is *FileName*, the file name is given by the third argument *FileName*, followed by the extension *.SBF*, *.NMA* or *.ECM* for SBF, NMEA and user-message files respectively. User-message files contain messages entered by the command **exeEchoMessage** prefixed with the GPS week number and time of week in seconds.

If *NamingType* is *Incremental*, the file name is given by the first five characters of the *FileName* argument (right padded with "_" if necessary), followed by a modulo-1000 counter incrementing each time logging is stopped and restarted. The file name extension is *.SBF*, *.NMA* or *.ECM* as described above. If the auto-incremented file name already exists on the disk, the receiver takes action as specified by the **setDiskFullAction** command.

If *NamingType* is *IGS15M*, *IGS1H*, *IGS6H* or *IGS24H*, the receiver automatically creates a new file every 15 minutes, every hour, every 6 hours or every 24 hours respectively, and the file name adheres to the IGS/RINEX naming convention. The 4-character station name is taken from the marker name as set by the **setMarkerParameters** command.

In IGS naming mode, the files are put in daily directories, the directory name being of the form *yyddd* with *yy* the 2-digit year and *ddd* the day of year. If *NamingType* is *FileName* or *Incremental*, the file is put in the root directory.

The set of allowed characters for the *FileName* argument is:

_0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

Note that the actual file name on the disk is case insensitive and only contains lower-case characters even if the user entered upper-case characters in the *FileName* argument.

If internal logging is ongoing at the moment when the command is entered, the current file is closed and the logging continues in a new file with the name as specified.

Examples

To have a fixed file name "mytest.sbf", use:

```
COM1> sfn, all, FileName, mytest <CR>
$R: sfn, all, FileName, mytest
      FileNaming, DSK1, FileName, "mytest"
COM1>
```

To create a new SBF file every hour on DSK1 with a filename according to the IGS convention, use:

```
COM1> sfn, DSK1, IGS1H <CR>  
$R: sfn, DSK1, IGS1H  
    FileNaming, DSK1, IGS1H, "mytest"  
COM1>
```

sfpr gfpr	setFTPPushRINEX getFTPPushRINEX	Server (32)	Path (64)	User (12)	Password (24)					
				anonymous						

[RxControl: Logging > Internal RINEX Logging > RINEX FTP Push Options](#)
[Web Interface: Logging > Internal RINEX Logging > RINEX FTP Push Options](#)

Use this command to automatically send the onboard RINEX files to a remote FTP server (FTP Push). The arguments specify the FTP server hostname or IP address, the path to the remote directory where to put the RINEX files, and the login and password to use. Note that the receiver encrypts the password so that it cannot be read back with the command **getFTPPushRINEX**.

The RINEX files are FTPed when they are complete, as prescribed by the *FileDuration* settings in the **setRINEXLogging** command. The current files are also FTPed when the user disables RINEX logging.

The files are put in the remote directory specified in the *Path* argument. Special character sequences can be used to encode the file date in the path: %y is replaced with the 2-digit year, %Y with the 4-digit year, %m with the month, %d with the day of the month, and %j with the day of the year (starting with 001). To put a literal "%" in the path, use %%. After expansion, *Path* must not be longer than 80 characters.

If the directory specified in the *Path* argument does not exist on the remote server, it is created.

If the transfer or the directory creation fails, an error is flagged (enter the command **lstInternalFile**, **Error** to see the errors and clear the error flag).

Example

```
COM1> sfpr, ftp.mydomain.com, mydata/%Y%m%d, myname, mypwd <CR>
$R: sfpr, ftp.mydomain.com, mydata/%Y%m%d, myname, mypwd
    FTPPushRINEX, "ftp.mydomain.com", "mydata/%Y%m%d",
    "myname", "7UU5CL7W1C75DWXX2TEXD3W"
COM1>
```

sfps	setFTPPushSBF	Server (32)	Path (64)	User (12)	Password (24)					
gfps	getFTPPushSBF			anonymous						

[RxControl: Logging > Internal Logging Settings](#)

[Web Interface: Logging > Internal Logging Settings](#)

Use this command to automatically send the onboard SBF files to a remote FTP server (FTP push). The arguments specify the FTP server hostname or IP address, the path to the remote directory where to put the SBF files, and the login and password to use. Note that the receiver encrypts the password so that it cannot be read back with the command **getFTPPushSBF**.

FTP push is only available in IGS file naming mode (see the **setFileNaming** command). Each time a SBF file is ready, it is FTPed to the specified server. For example, in **IGS1H** file naming mode, files are FTPed every hour. The current log file is also FTPed when the user disables internal logging.

The files are put in the remote directory specified in the *Path* argument. Special character sequences can be used to encode the file date in the path: %y is replaced with the 2-digit year, %Y with the 4-digit year, %m with the month, %d with the day of the month, and %j with the day of the year (starting with 001). To put a literal "%" in the path, use %%. After expansion, *Path* must not be longer than 80 characters.

If the directory specified in the *Path* argument does not exist on the remote server, it is created.

If the transfer or the directory creation fails, an error is flagged (enter the command **lstInternalFile**, **Error** to see the errors and clear the error flag).

Example

```
COM1> sfps, ftp.mydomain.com, mydata/%Y%m%d, myname, mypwd <CR>
$R: sfps, ftp.mydomain.com, mydata/%Y%m%d, myname, mypwd
    FTPPushSBF, "ftp.mydomain.com", "mydata/%Y%m%d",
    "myname", "7UU5CL7W1C75DWXX2TEXD3W"
COM1>
```


emd gmd	exeManageDisk getManageDisk	Disk	Action							
		DSK1	Unmount Format							

[RxControl: Logging > Disk Management](#)
[Web Interface: Logging > Disk Management](#)

Use this command to manage the internal disk identified by *Disk*.

Specify the action `Format` to format the disk (all data will be lost).

With the action `Unmount`, you command the receiver to stop all internal logging and to cleanly unmount the disk. After unmounting the disk, it is safe to power-off the receiver without danger of file corruption. Be aware that the only way to remount the disk is to reset or power-cycle the receiver. Note that the disk is also automatically unmounted when issuing the command `exePowerMode, standby`.

Prior to formatting or unmounting the disk, make sure to stop all disk activity such as logging, file listing or FTP download from the disk. If the specified action could not be performed, an error message is returned.

Example

To format the first internal disk `DSK1`, use:

```
COM1> emd, DSK1, Format <CR>
$R: emd, DSK1, Format
      ManageDisk, DSK1, Format
COM1>
```

lrf	1stRecordedFile	Disk	FileName (60)							
		DSK1								

Use this command to retrieve the contents of one of the internal log files.

The reply to this command consists in a succession of blocks starting with the "\$-- BLOCK" header, and terminating with the pseudo-prompt "---->" (see section Section 2.2.2, "Command Replies" for details). The decoding program must remove these headers and pseudo-prompts to recover the original file contents.

The download speed is highly influenced by the processor load. To speed up the download, it is recommended to stop the signal tracking, which can be done by typing the following command before starting the download: **setSatelliteTracking, none**. It is also recommended to perform file download over USB if speed is important.

The file download can be interrupted by sending ten uppercase "S" characters (simply by holding the "shift-S" key pressed) to the connection through which the download is taking place.

Examples

To output the contents of the internal log file named `log.sbf` on the first internal disk (DSK1), use:

```
COM1> lrf, DSK1, log.sbf <CR>
$R; lrf, DSK1, log.sbf
... Here comes the content of log.sbf ...
COM1>
```

If the file `log.sbf` does not exist, an error is returned:

```
COM1> lrf, DSK1, log.sbf <CR>
$R? 1stRecordedFile: Argument 'FileName' could not be handled!
COM1>
```

erf	exeRemoveFile	Disk	FileName (60)							
grf	getRemoveFile									
		DSK1	none							
			all							

[RxControl: Logging > Remove Internal File](#)

[Web Interface: Logging > Remove Internal File](#)

Use this command to remove one file or an entire directory from the internal disk identified by *Disk*.

If *FileName* is the name of a file, only that single file is removed from the disk. Files in a directory can be specified using dirname/filename.

If *FileName* is the name of a directory, the entire directory is deleted, except the file currently written to, if any.

If the reserved string `all` is used for the *FileName* argument, all files are removed from the selected disk, except the file currently written to, if any.

If there is no file nor directory named *FileName* on the disk or if the file is currently written to, an error message is returned.

Examples

To remove the file "ATRX2980.03_" from directory "03298", use:

```
COM1> erf, DSK1, 03298/ATRX2980.03_ <CR>
$R: erf, DSK1, 03298/ATRX2980.03_
      RemoveFile, DSK1, "03298/ATRX2980.03_"
COM1>
```

To remove all files from DSK1, use:

```
COM1> erf, DSK1, all <CR>
$R: erf, DSK1, all
      RemoveFile, DSK1, all
COM1>
```

srxi grxi	setRINEXLogging getRINEXLogging	Cd Cd	FileDuration	ObsInterval	SignalTypes	ExtraObsTypes	RINEXVersion	MixedNav		
		+ DSK1 all	none hour1 hour6 hour24 minute15	sec1 sec2 sec5 sec10 sec30 sec60	none + GPSL1CA + GPSL1PY + GPSL2PY + GPSL2C + GPSL5 + GLOL1CA + GLOL2P + GLOL2CA + GLOL3 + GALL1BC + GALE5a + GALE5b + GALE5 + GEOL1 + GEOL5 + CMPL1 + CMPE5b + QZSL1CA + QZSL2C + QZSL5 all	none + Dx + Sx all	v2x v3x	off on		

[RxControl: Logging > Internal RINEX Logging > RINEX Logging Options](#)
[Web Interface: Logging > Internal RINEX Logging > RINEX Logging Options](#)

Use this command to configure the RINEX files logged by the receiver.

The argument *Cd* specifies where the RINEX files should be logged. `DSK1` is the internal SD memory card.

The argument *FileDuration* specifies whether a new RINEX file should be started every 15 minutes, every hour, 6 hours or every day. When *FileDuration* is set to `none`, RINEX logging is disabled and all following arguments are ignored.

ObsInterval specifies the interval of the observation records.

SignalTypes sets the list of signals to encode in RINEX. The more signals are selected, the bigger the RINEX file.

By default, the RINEX file contains the code and carrier phase observables. It is possible to also include the Doppler (obs code Dx) and the C/N0 observables (obs code Sx) with the *ExtraObsTypes* argument.

The argument *RinexVersion* selects which RINEX version to use.

The argument *MixedNav* specifies whether the navigation data is stored in separate files for each constellation (*MixedNav* set to `off`), or in a single mixed file (*MixedNav* set to `on`). This argument is ignored if *RINEXVersion* is `v2x`, as RINEX v2.x does not allow mixed navigation files. Note that QZSS and Compass/BeiDou navigation data are only available in the mixed navigation file.

The RINEX file name complies with the RINEX v3.01 naming convention. The 4-character station name is taken from the marker name as set by the **setMarkerParameters** command. RINEX files are put in daily directories, the directory name being of the form `yyddd` with `yy` the 2-digit year and `ddd` the day of year.

If a RINEX file is currently being logged when issuing this command, the new settings will only be applied when the next RINEX file will be started. This occurs at a rate specified by

FileDuration. To force the new settings to be immediately applied, RINEX logging must be temporarily stopped (*FileDuration* set to `none`) and then re-enabled. Changing the RINEX settings (e.g. changing the list of signals to be stored in RINEX) results in the past data to be overwritten in the RINEX file.

Examples

To create daily RINEX files with the observation file containing only GPS L1CA data at a 30-s interval, use:

```
COM1> srxl, DSK1, hour24, sec30, GPSL1CA <CR>
$R: srxl, DSK1, hour24, sec30, GPSL1CA
    RINEXLogging, DSK1, Hour24, sec30, GPSL1CA, none, v2x, on
COM1>
```

To stop RINEX logging:

```
COM1> srxl, DSK1, none <CR>
$R: srxl, DSK1, none
    RINEXLogging, DSK1, none, sec30, GPSL1CA, none, v2x, on
COM1>
```

2.3.12 L-Band Commands

lbb	lbb	lbb	lbb	lbb	lbb	lbb	lbb	lbb	lbb	lbb

Use this command to retrieve the list of user-defined and auto-defined L-Band beams.

The list contains user-defined beams (User1, User2,...) defined with the **setLBandBeams** command and service-specific beams (LBAS1|1, LBAS1|2,...) which are automatically updated by the L-Band service provider. Only the enabled user-defined beams are shown.

For each beam, the list contains the beam carrier frequency in Hz, the baud rate, the beam name and the region code. For service-specific beams, the satellite longitude in degrees (from -180 to 180, positive east of Greenwich) and the grant status are also provided.

This command is very similar to the command **getLBandBeams**, the only difference being that the latter only reports the list of user-defined beams.

Example

```
COM1> 1lbb <CR>
$R; 1lbb
---->
$-- BLOCK 1 / 1
LBandBeams, User1, 1535165000, baud1200, "User", "E", Enabled
LBandBeams, LBAS1|1, 1539982500, baud1200 , "AORE" , "A", -15.5, "
    Granted"
LBandBeams, LBAS1|2, 1539892500, baud1200 , "AORW" , "B", -54.0, "
    Granted"
...
COM1>
```

slbb glbb	setLBandBeams getLBandBeams	Beam Beam	Frequency	Rate	Name (8)	Region (8)	Usage			
		+ User1 ... User16 all	1525000000 ...1559000000 Hz	baud600 baud1200 baud2400 baud4800	Unknown	Unknown	Disabled Enabled			

RxControl: L-band > Generic L-Band Settings

Web Interface: Configuration > L-band > Generic L-Band Settings

This command can be used to define/inquire the parameters of user-defined L-Band beams. A beam is characterized by its frequency and baud rate (the *Freq* and *Rate* arguments). Optionally, a beam name and region ID can also be associated to each beam, for information only. A beam can be enabled or disabled, as set by the *Usage* argument. Only enabled beams can be locked to.

Example

```
COM1> slbb, User1, 1537460000, baud1200, 25East, E, Enabled <CR>
$R: slbb, User1, 1537460000, baud1200, 25East, E, Enabled
    LBandBeams, User1, 1537460000, baud1200, "25East", "E", Enabled
COM1>
```


sism gism	setLBandSelectMode getLBandSelectMode	Mode	Service	Beam						
		auto	LBAS1	User1						
		off		User2 ... User16						
		manual		LBAS1 1						
				LBAS1 2						
				LBAS1 3						
				LBAS1 4						
				LBAS1 5						
				LBAS1 6						
				LBAS1 7						
				LBAS1 8						
				LBAS1 9						
				LBAS1 10						
				LBAS1 11						
				LBAS1 12						
				LBAS1 13						
				LBAS1 14						
				LBAS1 15						
				LBAS1 16						

[RxControl: L-band > Generic L-Band Settings](#)

[Web Interface: Configuration > L-band > Generic L-Band Settings](#)

This command can be used to define/inquire the main operation mode of the L-Band demodulator.

The following modes are available through the *Mode* argument:

Mode	Description
auto	The demodulator will try to lock to a visible beam, preferring beams to which access has been granted. The list of beams and their status can be retrieved by the command 1stLBandBeams .
off	The demodulator will be disabled and will not attempt to lock to any beam.
manual	The demodulator will attempt to lock to the beam identified in the <i>Beam</i> argument and ignore all other beams. The parameters of the beams (frequency and baud rate) can be retrieved by the command 1stLBandBeams . Make sure that the beam identified in the <i>Beam</i> argument is enabled (see command setLBandBeams).

The second argument *Service* specifies which service the demodulator has to lock to. Only the LBAS1 service is supported.

Example

```
COM1> sism, manual, LBAS1, User1 <CR>
$R: sism, manual, LBAS1, User1
    LBandSelectMode, manual, LBAS1, User1
COM1>
```

slpc	setLBAS1PPPConfig	Source								
glpc	getLBAS1PPPConfig									
		ULTRA								
		APEX								

RxControl: Navigation > Positioning Mode > PPP and Differential Corrections

Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections

This LBAS1-specific command configures the PPP positioning engine to use PPP corrections of the type specified by the *Source* argument.

This command is only available if the "augm_data_svc" permission is set to "MBAS1 default provider for global use". Use the command **lstInternalFile, Permissions** to check your permission file.

Example

```
COM1> slpc, APEX <CR>
$R: slpc, APEX
LBAS1PPPConfig, APEX
COM1>
```

llrs	lstLBAS1RefStations									

Use this LBAS1-specific command to inquire the list of the reference stations of which corrections are available in the LBAS1 L-Band beam currently locked to. Usage of the corrections from a given reference station can be granted or not, as indicated in the list.

Example

```
COM1> llrs <CR>
$R; llrs
---->
$-- BLOCK 1 / 1
LBAS1ReferenceStation, 0001, granted
LBAS1ReferenceStation, 0002, granted
LBAS1ReferenceStation, 0003, granted
LBAS1ReferenceStation, 0004, granted
LBAS1ReferenceStation, 0005, granted
LBAS1ReferenceStation, 0007, granted
LBAS1ReferenceStation, 0009, granted
LBAS1ReferenceStation, 0015, granted
LBAS1ReferenceStation, 0125, denied
LBAS1ReferenceStation, 0556, denied
...
COM1>
```

slrs glrs	setLBAS1RefStations getLBAS1RefStations	Stream Stream	StationID (255)							
		+ RTCMV all	all							

[RxControl: L-band > LBAS1-Specific Settings > Reference Stations](#)

[Web Interface: Configuration > L-band > LBAS1-Specific Settings > Reference Stations](#)

This LBAS1-specific command defines the set of reference stations from which differential corrections have to be demodulated from the L-Band beam and included in the decoded correction stream. Currently, only one decoded stream is defined, the RTCMV stream. That stream is fed into the PVT algorithm and serves as source of differential corrections in DGPS-rover and in PPP positioning modes.

The argument *StationID* is a list of 4-digit reference station IDs, separated by the "+" or "-" sign. Use the keyword "none" to empty the selected decoded correction stream, and "all" to include all reference stations in the stream.

Examples

To restrain the decoded differential correction stream to reference stations 1, 2, 3 and 15, use:

```
COM1> slrs, RTCMV, 0001+0002+0003+0015 <CR>
$R: slrs, RTCMV, 0001+0002+0003+0015
    LBAS1RefStations, RTCMV, "0001+0002+0003+0015"
COM1>
```

To add reference station 4 to the set, use:

```
COM1> slrs, RTCMV, +0004 <CR>
$R: slrs, RTCMV, +0004
    LBAS1RefStations, RTCMV, "0001+0002+0003+0004+0015"
COM1>
```

To select all reference stations except the one with ID 0015, use:

```
COM1> slrs, RTCMV, all-0015 <CR>
$R: slrs, RTCMV, all-0015
    LBAS1RefStations, RTCMV, "all-0015"
COM1>
```

To empty the RTCMV stream, use:

```
COM1> slrs, RTCMV, none <CR>
$R: slrs, RTCMV, none
    LBAS1RefStations, RTCMV, "none"
COM1>
```

spas	setPPPAutoSeed	Mode								
gpas	getPPPAutoSeed									
		none								
		+ DGPS								
		+ RTKFixed								
		all								

RxControl: Navigation > Positioning Mode > PPP and Differential Corrections

Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections

Use this command to specify which position mode is allowed to be used as a seed for the PPP engine.

If both **RTKFixed** and **DGPS** modes are enabled, the receiver gives priority to **RTKFixed** seeding over **DGPS** seeding.

In any case, a manual seed entered with the command **exePPPSetSeedGeod** overrules any automatic seeding.

Before enabling seeding from DGNSS or RTK, make sure that the DGNSS/RTK datum is specified with the **setGeodeticDatum** command, or alternatively that the offset between ITRF and the datum to which DGNSS and RTK positions relate is specified with the command **setPPPDatumOffset**.

Example

```
COM1> spas, RTKFixed <CR>
$R: spas, RTKFixed
PPPAutoSeed, RTKFixed
COM1>
```

spdo gpdo	setPPPDatumOffset getPPPDatumOffset	Mode	DX	DY	DZ					
		manual	-1000.000 ... 0.000 ... 1000.000 m	-1000.000 ... 0.000 ... 1000.000 m	-1000.000 ... 0.000 ... 1000.000 m					

RxControl: Navigation > Positioning Mode > PPP and Differential Corrections

Web Interface: Configuration > Navigation > Positioning Mode > PPP and Differential Corrections

If no transformation is applied, PPP positions relate to the ITRFyy datum (yy depending on the PPP service provider), which differs from the regional datum in which DGNSS or RTK positions are computed. In situations where the receiver can switch between DGNSS/RTK and PPP positioning modes, this means that the coordinates reported in SBF would exhibit a jump at each transition from and to PPP mode.

The preferred way to avoid this so-called datum shift is to use the **setGeodeticDatum** command to tell the receiver to transform all PPP coordinates to the regional DGNSS/RTK datum.

The **setPPPDatumOffset** command offers an additional way of removing the datum shift by instructing the receiver to subtract the values (DX, DY, DZ) from all PPP coordinates. This is done prior to the transformation to the datum specified by the **setGeodeticDatum** command.

The command has also an effect on seeding. In manual seeding mode (see the **exePPPSetSeedGeod** command), the receiver first performs the coordinate transformation from the datum specified by the *Datum* argument of the **exePPPSetSeedGeod** command, and then adds the offset (DX, DY, DZ) to the transformed coordinates. In auto-seed mode (see the **setPPPAutoSeed** command), the receiver first transforms the DGNSS/RTK coordinates according to the datum set by the **setGeodeticDatum** command and then adds the offset (DX, DY, DZ) to the transformed coordinates.

In most cases, it is recommended to use the **setGeodeticDatum** command to deal with datum offsets. The command **setPPPDatumOffset** is kept for backwards compatibility reasons though.

Example

```
COM1> spdo, manual, 0.22, 0.12, 0.54 <CR>
$R: spdo, manual, 0.22, 0.12, 0.54
    PPPDatumOffset, manual, 0.220, 0.120, 0.540
COM1>
```

2.3.13 SBF List

ASCIIIn	AttCovEuler	AttEuler
BBSamples	BaseLine	BaseStation
BaseVectorCart	BaseVectorGeod	CMPNav
CMPRaw	ChannelStatus	Commands
Comment	DOP	DiffCorrIn
DiskStatus	EndOfAtt	EndOfMeas
EndOfPVT	ExtEvent	ExtEventPVTCartesian
ExtEventPVTGeodetic	GALAlm	GALGstGps
GALlon	GALNav	GALRawFNAV
GALRawINAV	GALSARRLM	GALUtc
GEOAlm	GEOClockEphCovMatrix	GEOCorrections
GEODegrFactors	GEOFastCorr	GEOFastCorrDegr
GEOIGPMask	GEOIntegrity	GEOlonoDelay
GEOLongTermCorr	GEOMT00	GEONav
GEONetworkTime	GEOPRNMask	GEORawL1
GEORawL5	GEOServiceLevel	GLOAlm
GLONav	GLORawCA	GLOTime
GPSAlm	GPSlon	GPSNav
GPSRawCA	GPSRawL2C	GPSRawL5
GPSUtc	Group1	Group2
Group3	Group4	IPStatus
IQCorr	ISMR	InputLink
LBAS1DecoderStatus	LBAS1Messages	LBandBeams
LBandTrackerStatus	MeasEpoch	MeasExtra
NTRIPClientStatus	OutputLink	PVTCartesian
PVTGeodetic	PVTResiduals	PVTSatCartesian
PVTSupport	PosCart	PosCovCartesian
PosCovGeodetic	PosLocal	PosProjected
QZSNav	QZSRawL1CA	QZSRawL2C
QZSRawL5	QualityInd	RAIMStatistics
RTCMDatum	ReceiverSetup	ReceiverStatus
ReceiverTime	SatVisibility	VelCovCartesian
VelCovGeodetic	xPPSOffset	

Chapter 3

SBF Block Interface

3.1 Introduction

3.1.1 Scope

This document describes the format of the binary data output by Septentrio receivers, called SBF.

3.1.2 Typographical Conventions

abc User command name;
abc SBF block name and field name.

3.1.3 Change Log

Date	Change Description
Feb 04, 2015	Added the <code>QZSNav</code> block containing decoded QZSS navigation data
Jan 13, 2015	Added the <code>PosProjected</code> block containing plane grid coordinates
Dec 12, 2014	Added the base measurements quality indicator
July 14, 2014	Added the <code>ISMR</code> block containing ionospheric scintillation parameters
April 30, 2014	Added new values for the <code>Datum</code> field
April 22, 2014	Added the <code>DiskStatus</code> block reporting the disk usage and free space of the disks available on the receiver
Feb 21, 2014	Added the <code>NTRIPClientStatus</code> block for the NTRIP client connection status
March 14, 2013	Added the <code>QualityInd</code> block containing various quality indicators
Feb 19, 2013	Added the <code>CMPNav</code> block containing decoded Compass/BeiDou navigation data
Feb 8, 2013	Fixed typo: field <code>t_oG</code> of <code>GALGstGps</code> changed to type <code>u4</code> and units of seconds
Jan 8, 2013	Added fields <code>HAccuracy</code> , <code>VAccuracy</code> and <code>Misc</code> to the <code>PVTCartesian</code> and <code>PVTGeodetic</code> blocks
Dec 19, 2012	Added PRNs 139 and 140 to the list of SBAS satellites (see section 2.9)
Oct 25, 2012	Added <code>RTCMDatum</code> and <code>PosLocal</code> blocks
Oct 19, 2012	Added <code>GEORawL5</code> block
Oct 1, 2012	Added new signal type for L-band and SBAS L5 signals (value 23 and 25), see section 2.10
Sep 29, 2012	Added <code>LBandBeams</code> block and added <code>SVID</code> field to <code>LBandTrackerStatus</code> block
Sep 20, 2012	Added field <code>PPPInfo</code> to the <code>PVTCartesian</code> and <code>PVTGeodetic</code> blocks
Jun 27, 2012	Added fields to the <code>LBAS1DecoderStatus</code> block to report various service subscription parameters

Feb 28, 2012	Added <code>GALSARLM</code> block
Feb 6, 2012	Added QZSS signals and <code>QZSRawL1CA</code> , <code>QZSRawL2C</code> and <code>QZSRawL5</code> blocks

3.2 SBF Outline

SBF is the binary output format of Septentrio receivers. In this format, the data are arranged in binary blocks referred to as SBF blocks.

Each SBF block consists of a sequence of numeric or alphanumeric fields of different types and sizes. The total block size is always a multiple of 4 bytes.

The fields of an SBF block may have one of the following types:

Type	Description
u1	Unsigned integer on 1 byte (8 bits)
u2	Unsigned integer on 2 bytes (16 bits)
u4	Unsigned integer on 4 bytes (32 bits)
i1	Signed integer on 1 byte (8 bits)
i2	Signed integer on 2 bytes (16 bits)
i4	Signed integer on 4 bytes (32 bits)
f4	IEEE float on 4 bytes (32 bits)
f8	IEEE float on 8 bytes (64 bits)
c1[X]	String of X ASCII characters, right padded with bytes set to 0 if needed.

Each multi-byte binary type is transmitted as little-endian, meaning that the least significant byte is the first one to be transmitted by the receiver. Signed integers are coded as two's complement.

Every SBF block begins with an 8-byte block header, which is followed by the block body.

3.2.1 SBF Block Header Format

Every SBF block starts with an 8-byte header having the following contents:

Parameter	Type	Description
Sync	c1[2]	The Sync field is a 2-byte array always set to 0x24, 0x40. The first byte of every SBF block has hexadecimal value 24 (decimal 36, ASCII '\$'). The second byte of every SBF block has hexadecimal value 40 (decimal 64, ASCII '@'). These two bytes identify the beginning of any SBF block and can be used for synchronization.
CRC	u2	The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2	The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: bits 0-12: block number; bits 13-15: block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6).
Length	u2	The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.

3.2.2 List of SBF Block Names and Numbers

The structure and contents of an SBF block are unambiguously identified by the block ID. For easier readability, a block name is also defined for each block. When invoking the **setSBFOutput** command to enable a given block, the block name should be specified.

The following table provides the list of the SBF blocks names and numbers available on the PolaRxS receiver, and a short description of the associated contents. The block number is contained in bits 0 to 12 of the block ID field (see section 3.2.1).

The "Flex Rate" column indicates whether a given block can be output at a user-defined rate and the "esoc" column whether it can be used as an argument of the **exeSBFOnce** command (see also section 3.2.8). The "Time stamp" column indicates which type of time is encoded in the block time stamp (see section 3.2.3 for details).

Block name	Block No	Content description	Flex Rate	esoc	Time Stamp
Measurement Blocks					
MeasEpoch	4027	measurement set of one epoch	•	•	R
MeasExtra	4000	additional info such as observable variance	•	•	R
IQCorr	4046	real and imaginary post-correlation values	•	•	R
ISMR	4086	ionospheric scintillation monitor (ISMR) data			R
EndOfMeas	5922	measurement epoch marker	•	•	R
Navigation Page Blocks					
GPSRawCA	4017	GPS CA navigation subframe			S
GPSRawL2C	4018	GPS L2C navigation frame			S
GPSRawL5	4019	GPS L5 navigation frame			S
GLORawCA	4026	GLONASS CA navigation string			S
GALRawFNAV	4022	Galileo F/NAV navigation page			S
GALRawINAV	4023	Galileo I/NAV navigation page			S
GEORawL1	4020	SBAS L1 navigation message			S
GEORawL5	4021	SBAS L5 navigation message			S
CMPRaw	4047	Compass/BeiDou navigation page			S
QZSRawL1CA	4066	QZSS L1 CA navigation frame			S
QZSRawL2C	4067	QZSS L2C navigation frame			S
QZSRawL5	4068	QZSS L5 navigation frame			S
GPS Decoded Message Blocks					
GPSNav	5891	GPS ephemeris and clock		•	S
GPSAlm	5892	Almanac data for a GPS satellite		•	S
GPSIon	5893	Ionosphere data from the GPS subframe 5		•	S
GPSUtc	5894	GPS-UTC data from GPS subframe 5		•	S
GLONASS Decoded Message Blocks					
GLONav	4004	GLONASS ephemeris and clock		•	S
GLOAlm	4005	Almanac data for a GLONASS satellite		•	S
GLOTime	4036	GLO-UTC, GLO-GPS and GLO-UT1 data		•	S
Galileo Decoded Message Blocks					
GALNav	4002	Galileo ephemeris, clock, health and BGD		•	S
GALAlm	4003	Almanac data for a Galileo satellite		•	S
GALIon	4030	NeQuick Ionosphere model parameters		•	S
GALUtc	4031	GST-UTC data		•	S
GALGstGps	4032	GST-GPS data		•	S
GALSARRLM	4034	Search-and-rescue return link message			S
Compass/BeiDou Decoded Message Blocks					
CMPNav	4081	Compass/BeiDou ephemeris and clock		•	S

Block name	Block No	Content description	Flex Rate	esoc	Time Stamp
QZSS Decoded Message Blocks					
QZSNav	4095	QZSS ephemeris and clock		•	S
SBAS Decoded Message Blocks					
GEOMT00	5925	MT00 : SBAS Don't use for safety applications			S
GEOPRNMask	5926	MT01 : PRN Mask assignments			S
GEOFastCorr	5927	MT02-05/24: Fast Corrections			S
GEOIntegrity	5928	MT06 : Integrity information			S
GEOFastCorrDegr	5929	MT07 : Fast correction degradation factors			S
GEONav	5896	MT09 : SBAS navigation message		•	S
GEODegrFactors	5930	MT10 : Degradation factors			S
GEONetworkTime	5918	MT12 : SBAS Network Time/UTC offset parameters			S
GEOAlm	5897	MT17 : SBAS satellite almanac		•	S
GEOIGPMask	5931	MT18 : Ionospheric grid point mask			S
GEOLongTermCorr	5932	MT24/25 : Long term satellite error corrections			S
GEOIonoDelay	5933	MT26 : Ionospheric delay corrections			S
GEOServiceLevel	5917	MT27 : SBAS Service Message			S
GEOClockEphCovMatrix	5934	MT28 : Clock-Ephemeris Covariance Matrix			S
Position, Velocity and Time Blocks					
PVTCartesian	4006	Position, velocity, and time in Cartesian coordinates	•	•	R
PVTGeodetic	4007	Position, velocity, and time in geodetic coordinates	•	•	R
PosCovCartesian	5905	Position covariance matrix (X,Y, Z)	•	•	R
PosCovGeodetic	5906	Position covariance matrix (Lat, Lon, Alt)	•	•	R
VelCovCartesian	5907	Velocity covariance matrix (X, Y, Z)	•	•	R
VelCovGeodetic	5908	Velocity covariance matrix (North, East, Up)	•	•	R
DOP	4001	Dilution of precision	•	•	R
PosCart	4044	Position, variance and baseline in Cartesian coordinates	•	•	R
PosLocal	4052	Position in a local datum	•	•	R
PosProjected	4094	Plane grid coordinates	•	•	R
PVTSatCartesian	4008	Satellite positions	•	•	R
PVTResiduals	4009	Measurement residuals	•	•	R
RAIMStatistics	4011	Integrity statistics	•	•	R
GEOCorrections	5935	Orbit, Clock and pseudoranges SBAS corrections	•	•	R
BaseVectorCart	4043	XYZ relative position and velocity with respect to base(s)	•	•	R
BaseVectorGeod	4028	ENU relative position and velocity with respect to base(s)	•	•	R
PVTSupport	4076	Reserved for maintenance and support	•	•	R
EndOfPVT	5921	PVT epoch marker	•	•	R
GNSS Attitude Blocks					
AttEuler	5938	GNSS attitude expressed as Euler angles	•	•	R
AttCovEuler	5939	Covariance matrix of attitude	•	•	R
EndOfAtt	5943	GNSS attitude epoch marker	•	•	R
Receiver Time Blocks					
ReceiverTime	5914	Current receiver and UTC time	•	•	R
xPPSOffset	5911	Offset of the xPPS pulse with respect to GNSS time			R
External Event Blocks					
ExtEvent	5924	Time at the instant of an external event			E

Block name	Block No	Content description	Flex Rate	esoc	Time Stamp
ExtEventPVTCartesian	4037	Cartesian position at the instant of an event			E
ExtEventPVTGeodetic	4038	Geodetic position at the instant of an event			E
Differential Correction Blocks					
DiffCorrIn	5919	Incoming RTCM or CMR message			R
BaseStation	5949	Base station coordinates			R
RTCMDatum	4049	Datum information from the RTK service provider			R
L-Band Demodulator Blocks					
LBandTrackerStatus	4201	Status of the L-band signal tracking		•	R
LBAS1DecoderStatus	4202	Status of the LBAS1 L-band service			R
LBAS1Messages	4203	LBAS1over-the-air message			R
LBandBeams	4204	L-band satellite/beam information		•	R
Status Blocks					
ChannelStatus	4013	Status of the tracking for all receiver channels	•	•	R
ReceiverStatus	4014	Overall status information of the receiver	•	•	R
SatVisibility	4012	Azimuth/elevation of visible satellites	•	•	R
InputLink	4090	Statistics on input streams	•	•	R
OutputLink	4091	Statistics on output streams	•	•	R
NTRIPClientStatus	4053	NTRIP client connection status		•	R
IPStatus	4058	IP address, gateway and MAC address		•	R
QualityInd	4082	Quality indicators		•	R
DiskStatus	4059	Internal logging status		•	R
Miscellaneous Blocks					
ReceiverSetup	5902	General information about the receiver set-up		•	R
Commands	4015	Commands entered by the user		•	R
Comment	5936	Comment entered by the user		•	R
BBSamples	4040	Baseband samples			E
ASCIIIn	4075	Search-and-rescue return link message			R
Deprecated or Obsolete Blocks					
BaseLine	5950				R

3.2.3 SBF Block Time Stamp (TOW and WNC)

Each SBF header is directly followed by a time stamp, which consists of two fields: TOW and WNC:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current GPS week.
WNC	u2	1 week	65535	The GPS week number associated with the TOW. WNC is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNC is also the Galileo week number plus 1024.

In the SBF time stamps, the definition of the week always follows the GPS convention even if the block contains data for another constellation. This means that WNC 0, TOW 0 corresponds to Jan 06,1980 at 00:00:00 UTC.

If the time-of-week or the week number is unknown, which is typically the case for a few seconds after start-up, the corresponding field is set to its Do-Not-Use value (see section

3.2.7). It does not mean that the SBF block is unusable, but simply that the receiver could not time-tag it. It is typical that the `TOW` field becomes valid before the `WNc` field.

The interpretation to give to the time stamp is block-dependent. Three types of time stamps are possible:

- *Receiver time stamp*: this type of time stamp is used for the SBF blocks containing synchronous data, i.e. data generated at a given epoch in the receiver time scale. Examples of such blocks are the measurement and PVT blocks (`MeasEpoch` and `PVTCartesian`). The time stamp is always a multiple of the output interval as specified by the `setSBFOutput` command (see also section 3.2.8). As soon as the receiver time is aligned with the GNSS time, the receiver time stamp is guaranteed to never decrease in successive SBF blocks.
- *SIS time stamp*: it is used for asynchronous blocks containing navigation message data from the signal-in-space. The time stamp corresponds to the time of reception of the end of the last navigation frame or page used to build the SBF block, rounded to the nearest multiple of the page duration. This time is expressed in the receiver time scale.
- *External time stamp*: this type of time stamp is used for SBF blocks triggered by external asynchronous events, such as the `ExtEvent` block.

For the blocks with a SIS or an external time stamp, there is no strict relation between the time stamp of the SBF blocks and their order of transmission. For example, the SBF stream may contain a `GPSNav` block with ephemeris parameters received one hour in the past (i.e. the time stamp is one hour in the past) followed by another block with a current receiver time stamp.

3.2.4 Sub-blocks

Some blocks contain sub-blocks. For example, the `PVTSatCartesian` block contains `NSatPos` sub-blocks, each sub-block containing data for one particular satellite. SBF blocks that contain sub-blocks also contain a `SBLength` field, which indicates the size of the sub-blocks in bytes.

3.2.5 Padding Bytes

Padding bytes are foreseen at the end of every SBF block body and sub-block, so that their total size is equal to `Length` or `SBLength` respectively. The padding bytes are just placeholders and should not be looked at by the decoding software. Their value is not defined.

3.2.6 SBF Revision Number

Each SBF block has an associated revision number. The revision number is incremented each time a backwards-compatible change is implemented.

As described in section 3.2.1, the block number is to be found in bits 0 to 12 of the `ID` field, and the revision is in bits 13 to 15 of that field.

A backwards-compatible change consists of adding one or more fields in the padding bytes, or in the fields marked as "reserved" in the block description. Such change should be unnoticed by properly written decoding software that ignore the contents of padding and reserved fields (see also section 3.2.12). Each time such change happens, the revision number is incremented. The revision at which a given field has been introduced is documented in the block description in chapter 3.3, unless that revision is 0 (see the `ReceiverSetup` block as an example). It is guaranteed that if a given field exists in revision N, it will also exist in all revisions after N: no fields are withdrawn from SBF.

3.2.7 Do-Not-Use Value

It might happen that one or more pieces of data in an SBF block are not known at block creation time. For example, when there are insufficient satellite measurements to compute a position solution, the position components found in the `X`, `Y` and `Z` fields of the `PVTCartesian` block will not be available. To indicate that a given data item is not available or is currently not provided by the receiver, the corresponding field is set to a 'Do-Not-Use' value that is never reached in normal operation.

When applicable, the Do-Not-Use value is mentioned in the block description. The Do-Not-Use value refers to the raw contents of the field, without applying the scale factor. A field set to its Do-Not-Use value should always be discarded by the decoding software.

3.2.8 Output Rate

In general, the default output rate for each SBF block is the renewal rate of the information. For instance, the `GPSNav` block is output each time a new ephemeris data set is received from a given GPS satellite. The default output rates of GNSS measurement blocks, PVT blocks and integrated INS/GNSS blocks depend on your permission set. These three rates can be checked by the command `getReceiverCapabilities`.

The default output rate is specified for each block in chapter 3.3. To instruct the receiver to output a given block at its default rate, the "OnChange" rate has to be specified in the `setSBFOutput` command. Note that the maximum rate actually available on your receiver may be lower than the one specified in chapter 3.3, depending on your permission set.

Some blocks can only be output at their default rate (e.g. the `GPSNav` block). Others can be decimated to a user-selectable rate (which is by nature lower than the default rate). A subset of blocks can also be output "once" using the `exeSBFOnce` command. This can be handy to get a one-shot overview of a particular receiver state. Whether a given block supports a user-selectable rate and whether it belongs to the "output once" set is indicated in the SBF block list in section 3.2.2.

Attempting to force another rate than the default one for those blocks that do not support a user-selectable rate has no effect.

3.2.9 Space Vehicle ID and GLONASS Frequency Number

Satellites are identified by the `SVID` (or `PRN`) and `FreqNr` fields, defined as follows:

Field	Type	Do-Not-Use	Description	RINEX satellite code
Value				
<code>SVID</code> or <code>PRN</code>	u1	0	<p>Satellite ID: The following ranges are defined:</p> <p>1-37: PRN number of a GPS satellite</p> <p>38-61: Slot number of a GLONASS satellite with an offset of 37 (R01 to R24)</p> <p>62: GLONASS satellite of which the slot number is not known</p> <p>63-68: Slot number of a GLONASS satellite with an offset of 38 (R25 to R30)</p> <p>71-102: PRN number of a GALILEO satellite with an offset of 70</p> <p>107-119: L-Band (MSS) satellite. Corresponding satellite name can be found in the <code>LBandBeams</code> block.</p> <p>120-140: PRN number of an SBAS satellite (S20 to S40)</p> <p>141-177: PRN number of a Compass/BeiDou satellite with an offset of 140</p> <p>181-187: PRN number of a QZSS satellite with an offset of 180</p> <p>191-197: PRN number of an IRNSS satellite with an offset of 190</p> <p>198-215: PRN number of an SBAS satellite with an offset of 157 (S41 to S58)</p>	<p><i>Gnn</i> (<i>nn</i> = SVID)</p> <p><i>Rnn</i> (<i>nn</i> = SVID-37)</p> <p>NA</p> <p><i>Rnn</i> (<i>nn</i> = SVID-38)</p> <p><i>Enn</i> (<i>nn</i> = SVID-70)</p> <p>NA</p> <p><i>Snn</i> (<i>nn</i> = SVID-100)</p> <p><i>Cnn</i> (<i>nn</i> = SVID-140)</p> <p><i>Jnn</i> (<i>nn</i> = SVID-180)</p> <p><i>Inn</i> (<i>nn</i> = SVID-190)</p> <p><i>Snn</i> (<i>nn</i> = SVID-157)</p>
<code>FreqNr</code>	u1	0	<p>GLONASS frequency number, with an offset of 8. It ranges from 1 (corresponding to an actual frequency number of -7) to 21 (corresponding to an actual frequency number of 13).</p> <p>For non-GLONASS satellites, <code>FreqNr</code> is reserved and must be ignored by the decoding software.</p>	

3.2.10 Signal Type

Some sub-blocks contain a signal type field, which identify the type of signal and modulation the sub-blocks applies to. The signal numbering is defined as follows:

Signal number	Signal name	Carrier frequency (MHz)	RINEX v3.xx obs code
0	GPS_L1-CA	1575.42	1C
1	GPS_L1-P(Y)	1575.42	1W
2	GPS_L2-P(Y)	1227.60	2W
3	GPS_L2C	1227.60	2L
4	GPS_L5	1176.45	5Q
5	Reserved		
6	QZSS_L1-CA	1575.42	1C
7	QZSS_L2C	1227.60	2L
8	GLO_L1-CA	$1602.00 + (\text{FreqNr} - 8) * 9/16$, with FreqNr as defined in section 3.2.9.	1C
9	Reserved		
10	GLO_L2-P	$1246.00 + (\text{FreqNr} - 8) * 7/16$	2P
11	GLO_L2-CA	$1246.00 + (\text{FreqNr} - 8) * 7/16$	2C
12	GLO_L3	1202.025	3Q
13-14	Reserved		
15	IRNSS_L5	1176.45	5A
16	Reserved		
17	GAL_L1BC	1575.42	1C
18	Reserved		
19	GAL_E6BC	1278.75	6C
20	GAL_E5a	1176.45	5Q
21	GAL_E5b	1207.14	7Q
22	GAL_E5	1191.795	8Q
23	LBand (MSS)	L-band beam specific	NA
24	GEO_L1CA	1575.42	1C
25	GEO_L5	1176.45	5I
26	QZSS_L5	1176.45	5Q
27	Reserved		
28	CMP_L1 (Compass/BeiDou B1)	1561.098	1I
29	CMP_E5b (Compass/BeiDou B2)	1207.14	7I
30	CMP_B3 (Compass/BeiDou B3)	1268.52	6I
31	Reserved		

3.2.11 Channel numbering

Some blocks contain a reference to the receiver channel number. Channel numbering starts at one. The maximum value for the channel number depends on the receiver type.

3.2.12 Decoding of SBF Blocks

In order to decode an SBF block, one has to identify the block boundaries in the data stream coming from the receiver. This involves searching for the initial "\$@" characters that mark the beginning of each SBF block. Since the "\$@" sequence can occur in the middle of an SBF block as well, additional checking is needed to make sure that a given "\$@" is indeed the beginning of a block. The following procedure is recommended to decode SBF data stream.

1. Wait until the "\$@" character sequence appears in the data stream from the receiver. When it is found, go to point 2.

2. Read the next two bytes. It should be the block CRC. Store this value for future reference.
3. Read the next two bytes and store them in a buffer. It should be the block ID.
4. Read the next two bytes and append them to the buffer. It should be the `Length` field of the SBF block. It should be a multiple of 4. If not, go back to point 1.
5. Read the next (`Length-8`) bytes and append them to the buffer. Compute the CRC of the buffer. The computed CRC should be equal to the CRC stored at point 2. If not, go back to point 1, else a valid SBF block has been detected and can be interpreted by the reading software.
6. If the block number (bits 0 to 12 of the `ID` field decoded at point 3) is of interest to your application, decode the SBF block.
7. Go back to point 1 and search for the new occurrence of the "\$@" sequence after the end of the last byte of the block that was just identified.

To ensure compatibility with future upgrades of SBF, it is recommended that the decoding software observes the following rules:

- Only bits 0 to 12 of the `ID` field must be used to identify a block. Bits 13 to 15 represent the revision number.
- The lengths of SBF blocks and sub-blocks should not be considered constant and hard-coded in the decoding software. Instead, the decoding software must use the `Length` and `SBLength` fields encoded in the SBF block.
- Padding bytes should be ignored.
- Reserved fields and reserved bits in bit-fields should be ignored.

3.3 SBF Block Definitions

3.3.1 Measurement Blocks

MeasEpoch	Number: 4027
	"OnChange" interval: 10 ms

This block contains all the GNSS measurements (observables) taken at the time given by the `TOW` and `WNc` fields.

For each tracked signal, the following measurement set is available:

- the pseudorange
- the carrier phase
- the Doppler
- the C/N0
- the lock-time.

To decrease the block size, all the measurements from a given satellite are referenced to one master measurement set. For instance, the L2 pseudorange (C2) is not much different from the L1 pseudorange (C1), such that the difference between C2 and C1 is encoded, instead of the absolute value of C2.

This is done by using a two-level sub-block structure. All the measurements from a given satellite are stored in a `MeasEpochChannelType1` sub-block. The first part of this sub-block contains the master measurements, encoded as absolute values. The second part contains slave measurements, for which only the delta values are encoded in smaller `MeasEpochChannelType2` sub-blocks.

Every `MeasEpochChannelType1` sub-block contains a field "N2", which gives the number of nested `MeasEpochChannelType2` sub-blocks. If there is only one signal tracked for a given satellite, there are no slave measurements and N2 is set to 0.

Decoding is done as follows:

1. Decode the master measurements and the N2 value from the `MeasEpochChannelType1` sub-block.
2. If N2 is not 0, decode the N2 nested `MeasEpochChannelType2` sub-blocks.
3. Go back to 1 till the N1 `MeasEpochChannelType1` sub-blocks have been decoded.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N1	u1			Number of <code>MeasEpochChannelType1</code> sub-blocks in this <code>MeasEpoch</code> block.
SB1Length	u1	1 byte		Length of a <code>MeasEpochChannelType1</code> sub-block, excluding the nested <code>MeasEpochChannelType2</code> sub-blocks
SB2Length	u1	1 byte		Length of a <code>MeasEpochChannelType2</code> sub-block

Rev 1

CommonFlags	u1			<p>Bit field containing flags common to all measurements.</p> <p>Bit 0: Multipath mitigation: if this bit is set, multipath mitigation is enabled. (see the setMultipathMitigation command).</p> <p>Bit 1: Smoothing of code: if this bit is set, at least one of the code measurements are smoothed values (see setSmoothingInterval command).</p> <p>Bit 2: Carrier phase align: if this bit is set, the fractional part of the carrier phase measurements from different modulations on the same carrier frequency (e.g. GPS L2C and L2P) are aligned, i.e. multiplexing biases (0.25 or 0.5 cycles) are corrected. Aligned carrier phase measurements can be directly included in RINEX files. If this bit is unset, this block contains raw carrier phase measurements. This bit is always set in the current firmware version.</p> <p>Bit 3: Clock steering: this bit is set if clock steering is active (see setClockSyncThreshold command).</p> <p>Bit 4: Not applicable.</p> <p>Bits 5-7: Reserved</p>
CumClkJumps	u1	0.001 s		Cumulative millisecond clock jumps since startup, with an ambiguity of $k \cdot 256$ ms. For example, if two clock jumps of -1 ms have occurred since startup, this field contains the value 254.
Reserved	u1			Reserved for future use, to be ignored by decoding software
Type1		<i>A succession of N1 MeasEpochChannelType1 sub-blocks, see definition below</i>
Padding	u1[...]			Padding bytes, see 3.2.5

MeasEpochChannelType1 sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
RxChannel	u1			Receiver channel on which this satellite is currently tracked (see 3.2.11).
Type	u1			<p>Bit field indicating the signal type and antenna ID:</p> <p>Bits 0-4: signal number, see 3.2.10.</p> <p>Bits 5-7: Antenna ID: 0 for main, 1 for <i>Aux1</i> and 2 for <i>Aux2</i></p>
SVID	u1			Satellite ID, see 3.2.9
Misc	u1	4294967.296 m	0 ⁽¹⁾	<p>Bit field containing the MSB of the pseudorange.</p> <p>Bits 0-3: CodeMSB: MSB of the pseudorange (this is an unsigned value).</p> <p>Bits 4-7: Reserved</p>

⁽¹⁾ The pseudorange is invalid if both CodeMSB is 0 and CodeLSB is 0.

CodeLSB	u4	0.001 m	0 ⁽¹⁾	LSB of the pseudorange. The pseudorange expressed in meters is computed as follows: $PR_{type1}[m] = (CodeMSB * 4294967296 + CodeLSB) * 0.001$ where CodeMSB is part of the Misc field.
Doppler	i4	0.0001 Hz	-2147483648	Carrier Doppler (positive for approaching satellites). To compute the Doppler in Hz, use: $D_{type1}[Hz] = Doppler * 0.0001$
CarrierLSB	u2	0.001 cycles	0 ⁽²⁾	LSB of the carrier phase relative to the pseudorange
CarrierMSB	i1	65.536 cycles	-128 ⁽²⁾	MSB of the carrier phase relative to the pseudorange. The full carrier phase can be computed by: $L[cycles] = PR_{type1}[m] / \lambda + (CarrierMSB * 65536 + CarrierLSB) * 0.001$ where λ is the carrier wavelength corresponding to the frequency of the signal type in the Type field above: $\lambda = 299792458 / f_L$ m, with f_L the carrier frequency as listed in section 3.2.10.
CN0	u1	0.25 dB-Hz	255	The C/N0 in dB-Hz is computed as follows, depending on the signal type in the Type field: $C/N_0[dB-Hz] = CN0 * 0.25$ if the signal number is 1 or 2 $C/N_0[dB-Hz] = CN0 * 0.25 + 10$ otherwise
LockTime	u2	1 s	65535	Duration of continuous carrier phase. The lock-time is reset at the initial lock of the phase-locked-loop, and whenever a loss of lock condition occurs. If the lock-time is longer than 65534s, it is clipped to 65534s. If the carrier phase measurement is not available, this field is set to its Do-Not-Use value.
ObsInfo	u1		0	Bit field: Bit 0: if set, the pseudorange measurement is smoothed Bit 1: if set, the smoothing filter has reached the requested smoothing interval Bit 2: this bit is set when the carrier phase (L) has a half-cycle ambiguity Bits 3-7: FreqNr: for GLONASS satellites, these bits contain the frequency number with an offset of 8 (see 3.2.9), otherwise they are reserved and must be ignored by the decoding software.
N2	u1			Number of MeasEpochChannelType2 sub-blocks contained in this MeasEpochChannelType1 sub-block.
Padding	u1[.]			Padding bytes, see 3.2.5
Type2		A succession of N2 MeasEpochChannelType2 sub-blocks, see definition below

⁽²⁾ The carrier phase is invalid if both CarrierMSB is -128 and CarrierLSB is 0.

MeasEpochChannelType2 sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number, see 3.2.10. Bits 5-7: Antenna ID: 0 for main, 1 for <i>Aux1</i> and 2 for <i>Aux2</i>
LockTime	u1	1 s	255	See corresponding field in the MeasEpochChannelType1 sub-block above, except that the value is clipped to 254 instead of 65534.
CNO	u1	0.25 dB-Hz	255	See corresponding field in the MeasEpochChannelType1 sub-block above.
OffsetsMSB	u1	65.536 m 6.5536 Hz	-4 ⁽³⁾ -16 ⁽⁴⁾	Bit field containing the MSB of the code and of the Doppler offsets with respect to the MeasEpochChannelType1 sub-block. Bits 0-2: CodeOffsetMSB: MSB of the code offset. Bits 3-7: DopplerOffsetMSB: MSB of the Doppler offset. CodeOffsetMSB and DopplerOffsetMSB are coded as two's complement. Refer to the CodeOffsetLSB and DopplerOffsetLSB fields to see how to use this field.
CarrierMSB	i1	65.536 cycles	-128 ⁽⁵⁾	MSB of the carrier phase relative to the pseudorange.
ObsInfo	u1			Bit field: Bit 0: if set, the pseudorange measurement is smoothed Bit 1: if set, the smoothing filter has reached the requested smoothing interval Bit 2: this bit is set when the carrier phase (L) has a half-cycle ambiguity Bits 3-7: Reserved
CodeOffsetLSB	u2	0.001 m	0 ⁽³⁾	LSB of the code offset with respect to pseudorange in the MeasEpochChannelType1 sub-block. To compute the pseudorange, use: $PR_{type2}[m] = PR_{type1}[m] + (CodeOffsetMSB * 65536 + CodeOffsetLSB) * 0.001$
CarrierLSB	u2	0.001 cycles	0 ⁽⁵⁾	LSB of the carrier phase relative to the pseudorange. The full carrier phase can be computed by: $L[cycles] = PR_{type2}[m] / \lambda + (CarrierMSB * 65536 + CarrierLSB) * 0.001$ where λ is the carrier wavelength corresponding to the signal type in the Type field.

⁽³⁾ The pseudorange is invalid if both CodeOffsetMSB is -4 and CodeOffsetLSB is 0.

⁽⁴⁾ The Doppler is invalid if both DopplerOffsetMSB is -16 and DopplerOffsetLSB is 0.

⁽⁵⁾ The carrier phase is invalid if both CarrierMSB is -128 and CarrierLSB is 0.

DopplerOffsetLSB	u2	0.0001 Hz	0 ⁽⁴⁾	<p>LSB of the Doppler offset relative to the Doppler in the MeasEpochChannelType1 sub-block. To compute the Doppler, use:</p> $D_{type2}[\text{Hz}] = D_{type1}[\text{Hz}] * \alpha + (\text{DopplerOffsetMSB} * 65536 + \text{DopplerOffsetLSB}) * 1e-4,$ <p>where α is the ratio of the carrier frequency corresponding to the observable type in this MeasEpochChannelType2 sub-block, and that of the master observable type in the parent MeasEpochChannelType1 sub-block (see section 3.2.10 for a list of all carrier frequencies).</p>
Padding	u1[..]			Padding bytes, see 3.2.5

MeasExtra	Number: 4000 "OnChange" interval: 10 ms
-----------	--

This block contains extra information associated with the measurements contained in the MeasEpoch block, such as the internal corrections parameters applied during the measurement pre-processing, and the noise variances.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
N	u1			Number of sub-blocks in this MeasExtra block.
SBLength	u1	1 byte		Length of a sub-block
DopplerVarFactor	f4	1 Hz ² / cycle ²		Factor to be used to compute the Doppler variance from the carrier phase variance. More specifically, the Doppler variance in mHz ² can be computed by: $\sigma_{\text{Doppler}}^2[\text{mHz}^2] = \text{CarrierVariance} * \text{DopplerVarFactor},$ Where CarrierVariance can be found for each measurement type in the MeasExtraChannelSub sub-blocks.

<i>ChannelSub</i>		<i>A succession of $N_{MeasExtraChannelSub}$ sub-blocks, see definition below</i>
Padding	u1[...]			Padding bytes, see 3.2.5

MeasExtraChannelSub sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
RxChannel	u1			Receiver channel on which this satellite is currently tracked (see 3.2.11).
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number, see 3.2.10. Bits 5-7: Antenna ID: 0 for main, 1 for <i>Aux1</i> and 2 for <i>Aux2</i>
MPCorrection	i2	0.001 m		Multipath correction applied to the pseudorange. This number has to be added to the pseudorange to recover the raw pseudorange as it would be if multipath mitigation was not used.
SmoothingCorr	i2	0.001 m		Smoothing correction applied to the pseudorange. This number has to be added to the pseudorange to recover the raw pseudorange as it would be if smoothing was disabled.
CodeVar	u2	0.0001 m ²	65535	Estimated code tracking noise variance. If the variance is larger than 65534 cm ² , it is clipped to 65534 cm ² .
CarrierVar	u2	1 mcycle ²	65535	Estimated carrier tracking noise variance. This value can be multiplied by <i>DopplerVarFactor</i> to compute the Doppler measurement variance. If the variance is larger than 65534 mcycles ² , it is clipped to 65534 mcycles ² .
LockTime	u2	1 s	65535	Duration of continuous carrier phase. The lock-time is reset at the initial lock after a signal (re)acquisition. If the lock-time is longer than 65534s, it is clipped to 65534s. If the carrier phase measurement is not available, this field is set to its Do-Not-Use value.
CumLossCont	u1			Carrier phase cumulative loss-of-continuity counter for the signal type, antenna and satellite this sub-block refers to. This counter starts at zero at receiver start-up, and is incremented at each initial lock after signal (re)acquisition, or when a cycle slip is detected.
Reserved	u1			Reserved.
Info	u1			Bit field: Bits 0-3: Reserved Bits 4-7: Reserved.
Padding	u1[...]			Padding bytes, see 3.2.5

Rev 1

Rev 2

IQCorr	Number:	4046
	"OnChange" interval:	10 ms

This block contains punctual correlation values (real and imaginary parts) and carrier phase measurements (modulo 65.536 cycles) for all signal types except for GPS L2P and GLONASS L2P.

It is typical to output the `IQCorr` block at a 50-Hz or 100-Hz rate and the `MeasEpoch` block at 1-Hz or 10-Hz. The carrier phase measurement from the low-rate `MeasEpoch` block can be used to resolve the 65.536-cycle ambiguity of the carrier phase in the `IQCorr` block.

Note that high-rate output is only possible on USB or Ethernet connections. COM ports typically do not offer enough bandwidth to support 50-Hz `IQCorr` output.

Note that this feature may not be enabled on your receiver. It is under permission control.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Rev 1

N	u1			Number of sub-blocks in this IQCorr block.
SBLength	u1	1 byte		Length of a sub-block
CorrDuration	u1	0.001 s		Duration over which the correlations are computed (coherent integration time, except for SBAS L1 where a non-coherent integration is used).
CumClkJumps	u1	0.001 s		Cumulative millisecond clock jumps since start-up, with an ambiguity of $k \cdot 256$ ms. For example, if two clock jumps of -1 ms have occurred since startup, this field contains the value 254.
Reserved	u1[2]			Reserved for future use.
ChannelSub		A succession of N IQCorrChannelSub sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

IQCorrChannelSub sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
RxChannel	u1			Receiver channel on which this satellite is currently tracked (see 3.2.11).
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-5: signal number, see 3.2.10. Bits 6-7: Antenna ID: 0 for main, 1 for Aux1 and 2 for Aux2
SVID	u1			Satellite ID, see 3.2.9
CorrIQ_MSB	u1		136 ⁽⁶⁾	Bit field containing the MSB of the correlation values: Bits 0-3: I_MSB: MSB of the I correlation value, two's complement. See CorrI_LSB for usage. Bits 4-7: Q_MSB: MSB of the Q correlation value, two's complement. See CorrQ_LSB for usage.
CorrI_LSB	u1		0 ⁽⁶⁾	LSB of the real component of the punctual correlation value, unsigned. The full I correlation value is computed by: $I = I_MSB \cdot 256 + CorrI_LSB$
CorrQ_LSB	u1		0 ⁽⁶⁾	LSB of the imaginary component of the punctual correlation value, unsigned. The full Q correlation value is computed by: $Q = Q_MSB \cdot 256 + CorrQ_LSB$
CarrierPhaseLSB	u2	0.001 cycles		16-bit LSB of the carrier phase measurement, expressed in 0.001 cycles.
Padding	u1[...]			Padding bytes, see 3.2.5

Rev 1

⁽⁶⁾ The correlation values must be ignored if CorrIQ_MSB is set to 136 and CorrI_LSB is set to 0 and CorrQ_LSB is set to 0 (all conditions met together).

ISMR	Number: 4086
	"OnChange" interval: 60s

This block reports the S4 and the so-called "sigma phase" ionosphere scintillation parameters for all tracked satellites and signals. This block is output every minute on the minute.

S4 is the standard deviation of 50-Hz raw signal power samples normalized to the average signal power over an interval of 60 seconds.

Sigma phase is the standard deviation, in radians, of 50-Hz detrended carrier phase samples averaged over an interval of 60 seconds. It is also referred to as "Phi60". The detrending is performed by filtering the raw carrier phase measurements by a high-pass sixth order Butterworth filter having a cutoff frequency of 0.1Hz.

Note that this feature may not be enabled on your receiver. It is under permission control.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of sub-blocks in this <code>ISMR</code> block.

SBLength	u1	1 byte		Length of a sub-block
Reserved	u1[4]			Reserved for future use.
<i>ISMRChannel</i>		<i>A succession of N ISMRChannel sub-blocks, see definition below</i>
Padding	u1[..]			Padding bytes, see 3.2.5

ISMRChannel sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
RXChannel	u1			Receiver channel on which this satellite is currently tracked (see 3.2.11).
Type	u1			Signal type: Bits 0-5: signal number, see 3.2.10. Bits 6-7: Reserved
SVID	u1			Satellite ID, see 3.2.9
Reserved	u1			Reserved for future use.
S4	u2	0.001	65535	Amplitude scintillation index
SigmaPhi	u2	0.001 rad	65535	Phase scintillation index
Padding	u1[..]			Padding bytes, see 3.2.5

EndOfMeas	Number: 5922 "OnChange" interval: 10 ms
-----------	--

This block marks the end of the transmission of all measurement-related blocks belonging to a given epoch.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Padding	u1[...]			Padding bytes, see 3.2.5

3.3.2 Navigation Page Blocks

GPSSRawCA	Number: 4017 "OnChange" interval: 6s
-----------	---

This block contains the 300 bits of a GPS C/A subframe. It is generated each time a new subframe is received, i.e. every 6 seconds.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Not applicable
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved

FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			<p>NAVBits contains the 300 bits of a GPS C/A subframe.</p> <p>Encoding: For easier parsing, the bits are stored as a succession of 10 32-bit words. Since the actual words in the subframe are 30-bit long, two unused bits are inserted in each 32-bit word. More specifically, each 32-bit word has the following format:</p> <p>Bits 0-5: 6 parity bits (referred to as D_{25} to D_{30} in the GPS ICD), XOR-ed with the last transmitted bit of the previous word (D_{30}^*)).</p> <p>Bits 6-29: source data bits (referred to as d_n in the GPS ICD). The first received bit is the MSB.</p> <p>Bits 30-31: Reserved</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GPSTRawL2C	Number: 4018 "OnChange" interval: 12s
------------	--

This block contains the 300 bits of a GPS L2C CNAV subframe (the so-called $D_c(t)$ data stream).

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the subframe
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable

Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			<p>NAVBits contains the 300 bits of a GPS CNAV subframe.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[9] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GPSTRawL5	Number: 4019
	"OnChange" interval: 6s

This block contains the 300 bits of a GPS L5 CNAV subframe (the so-called $D_c(t)$ data stream).

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the subframe
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable

Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			<p>NAVBits contains the 300 bits of a GPS CNAV subframe.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[9] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GLORawCA	Number: 4026 "OnChange" interval: 2s
----------	---

This block contains the 85 bits of a GLONASS L1CA or L2CA navigation string.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <i>Sync1</i> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <i>Sync2</i> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <i>CRC</i> field is the 16-bit CRC of all the bytes in an SBF block from and including the <i>ID</i> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <i>ID</i> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <i>Length</i> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <i>TOW</i> . <i>WNc</i> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <i>WNc</i> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Not applicable
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Frequency number, with an offset of 8. See 3.2.9
Reserved	u1			Reserved for future use, to be ignored by decoding software.

NAVBits	u4[3]			<p>NAVBits contains the first 85 bits of a GLONASS C/A string (i.e. all bits of the string with the exception of the time mark).</p> <p>Encoding: The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[2] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GALRawFNAV	Number: 4022 "OnChange" interval: 10s
------------	--

This block contains the 244 bits of a Galileo F/NAV navigation page, after deinterleaving and Viterbi decoding.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the page
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable

Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			<p>NavBits contains the 244 bits of a Galileo F/NAV page.</p> <p>Encoding: NAVBits contains all the bits of the frame, with the exception of the synchronization field. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[7] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GALRawINAV	Number: 4023 "OnChange" interval: 2s
------------	---

This block contains the 234 bits of a Galileo I/NAV navigation page, after deinterleaving and Viterbi decoding.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the page

Source	u1			<p>Bit field:</p> <p>Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10</p> <p>Bit 5: Set when the nav page is the concatenation of a sub-page received from E5b, and a sub-page received from L1BC. In that case, bits 0-4 are set to L1BC.</p> <p>Bits 6-7: Reserved</p>
FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			<p>NAVBits contains the 234 bits of an I/NAV navigation page (in nominal or alert mode). Note that the I/NAV page is transmitted as two sub-pages (the so-called even and odd pages) of duration 1 second each (120 bits each). In this block, the even and odd pages are concatenated, even page first and odd page last. The 6 tails bits at the end of the even page are removed (hence a total of 234 bits). If the even and odd pages have been received from two different carriers (E5b and L1), bit 5 of the Source field is set.</p> <p>Encoding: NAVBits contains all the bits of the frame, with the exception of the synchronization field. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[7] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GEORawL1	Number: 4020 "OnChange" interval: 1s
----------	---

This block contains the 250 bits of a SBAS L1 navigation frame, after Viterbi decoding.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the navigation frame
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable

Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			<p>NAVBits contains the 250 bits of a SBAS navigation frame.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[7] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GEORawL5	Number: 4021 "OnChange" interval: 1s
----------	---

This block contains the 250 bits of a SBAS L5 navigation frame, after Viterbi decoding.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the navigation frame
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable

Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			<p>NAVBits contains the 250 bits of a SBAS navigation frame.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[7] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

CMPRaw	Number: 4047 "OnChange" interval: 6 seconds (non GEOs), 0.6 s (GEOs)
--------	---

This block contains the 300 bits of a Compass/BeiDou navigation page, as received from the CMP_L1 (B1), CMP_E5b (B2) or CMP_B3 (B3) signal.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Not applicable
Source	u1			Signal type from which the bits have been received, as defined in 3.2.10
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software.

NAVBits	u4[10]			<p>NAVBits contains the 300 deinterleaved bits of a Compass/BeiDou navigation subframe.</p> <p>Encoding: NAVBits contains all the bits of the subframe, including the preamble and the parity bits. The first received bit is stored as the MSB of NAVBits[0]. The 20 unused bits in NAVBits[9] must be ignored by the decoding software. The bits are deinterleaved.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

QZSRawL1CA	Number: 4066
	"OnChange" interval: 6s

This block contains the 300 bits of a QZSS C/A subframe.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
Reserved	u1			Reserved
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
Reserved2	u1[2]			Reserved for future use, to be ignored by decoding software.

NAVBits	u4[10]			NAVBits contains the 300 bits of a QZSS C/A subframe. Encoding: Same as GPSRawCA block.
Padding	u1[..]			Padding bytes, see 3.2.5

QZSRawL2C	Number: 4067 "OnChange" interval: 12s
-----------	--

This block contains the 300 bits of a QZSS L2C CNAV subframe.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the subframe
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software.

NAVBits	u4[10]			<p>NAVBits contains the 300 bits of a QZSS CNAV subframe.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[9] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

QZSRawL5	Number: 4068
	"OnChange" interval: 6s

This block contains the 300 bits of a QZSS L5 CNAV subframe.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			Satellite ID, see 3.2.9
CRCPassed	u1			Status of the CRC or parity check: 0: CRC or parity check failed 1: CRC or parity check passed
ViterbiCnt	u1			Viterbi decoder error count over the subframe
Source	u1			Bit field: Bits 0-4: Signal type from which the bits have been received, as defined in 3.2.10 Bits 5-7: Reserved
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software.

NAVBits	u4[10]			<p>NAVBits contains the 300 bits of a QZSS CNAV subframe.</p> <p>Encoding: NAVBits contains all the bits of the frame, including the preamble. The first received bit is stored as the MSB of NAVBits[0]. The unused bits in NAVBits[9] must be ignored by the decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.3 GPS Decoded Message Blocks

GPSTNav	Number: 5891 "OnChange" interval: block generated each time a new navigation data set is received from a GPS satellite
---------	---

The GPSTNav block contains the decoded navigation data for one GPS satellite. These data are conveyed in subframes 1 to 3 of the satellite navigation message. Refer to GPS ICD for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the GPS satellite of which the ephemeris is given in this block (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
WN	u2	1 week	65535	Week number (10 bits from subframe 1, word 3)
CAorPonL2	u1			Code(s) on L2 channel (2 bits from subframe 1, word 3)

URA	u1			User Range accuracy index (4 bits from subframe 1 word 3)
health	u1			6-bit health from subframe 1, word 3 (6 bits from subframe 1, word 3)
L2DataFlag	u1			Data flag for L2 P-code (1 bit from subframe 1, word 4)
IODC	u2			Issue of data, clock (10 bits from subframe 1)
IODE2	u1			Issue of data, ephemeris (8 bits from subframe 2)
IODE3	u1			Issue of data, ephemeris (8 bits from subframe 3)
FitIntFlg	u1			Curve Fit Interval, (1 bit from subframe 2, word 10)
Reserved2	u1			unused, to be ignored by decoding software
T_gd	f4	1 s		Estimated group delay differential
t_oc	u4	1 s		clock data reference time
a_f2	f4	1 s / s ²		SV clock aging
a_f1	f4	1 s / s		SV clock drift
a_f0	f4	1 s		SV clock bias
C_rs	f4	1 m		Amplitude of the sine harmonic correction term to the orbit radius
DEL_N	f4	1 semi-circle / s		Mean motion difference from computed value
M_0	f8	1 semi-circle		Mean anomaly at reference time
C_uc	f4	1 rad		Amplitude of the cosine harmonic correction term to the argument of latitude
e	f8			Eccentricity
C_us	f4	1 rad		Amplitude of the sine harmonic correction term to the argument of latitude
SQRT_A	f8	1 m ^{1/2}		Square root of the semi-major axis
t_oe	u4	1 s		Reference time ephemeris
C_ic	f4	1 rad		Amplitude of the cosine harmonic correction term to the angle of inclination
OMEGA_0	f8	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
C_is	f4	1 rad		Amplitude of the sine harmonic correction term to the angle of inclination
i_0	f8	1 semi-circle		Inclination angle at reference time
C_rc	f4	1 m		Amplitude of the cosine harmonic correction term to the orbit radius
omega	f8	1 semi-circle		Argument of perigee
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension
IDOT	f4	1 semi-circle / s		Rate of inclination angle
WNt_oc	u2	1 week		WN associated with t_oc, modulo 1024
WNt_oe	u2	1 week		WN associated with t_oe, modulo 1024
Padding	u1[..]			Padding bytes, see 3.2.5

GPSSAlm	Number: 5892
	"OnChange" interval: block generated each time a new almanac data set is received from a GPS satellite

The GPSSAlm block contains the decoded almanac data for one GPS satellite. These data are conveyed in subframes 4 and 5 of the satellite navigation message. Refer to GPS ICD for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the GPS satellite of which the almanac is given in this block (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
e	f4			Eccentricity
t_oa	u4	1 s		almanac reference time of week
delta_i	f4	1 semi-circle		Inclination angle at reference time, relative to $i_0 = 0.3$ semi-circles
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension

SQRT_A	f4	1 m ^{1/2}		Square root of the semi-major axis
OMEGA_0	f4	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
omega	f4	1 semi-circle		Argument of perigee
M_0	f4	1 semi-circle		Mean anomaly at reference time
a_f1	f4	1 s / s		SV clock drift
a_f0	f4	1 s		SV clock bias
WN_a	u1	1 week		Almanac reference week, to which t_oa is referenced
config	u1			Anti-spoofing and satellite configuration (4 bits from subframe 4, page 25)
health8	u1			health on 8 bits from the almanac page
health6	u1			health summary on 6 bits (from subframe 4, page 25 and subframe 5 page 25)
Padding	u1[..]			Padding bytes, see 3.2.5

GPSTime	Number: 5893
	"OnChange" interval: block generated each time subframe 4, page 18, is received from a GPS satellite

The GPSTime block contains the decoded ionosphere data (the Klobuchar coefficients). These data are conveyed in subframes 4, page 18 of the satellite navigation message. Refer to GPS ICD for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the GPS satellite from which the coefficients have been received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
alpha_0	f4	1 s		vertical delay coefficient 0
alpha_1	f4	1 s / semi-circle		vertical delay coefficient 1
alpha_2	f4	1 s / semi-circle ²		vertical delay coefficient 2
alpha_3	f4	1 s / semi-circle ³		vertical delay coefficient 3
beta_0	f4	1 s		model period coefficient 0

beta_1	f4	1 s / semi-circle		model period coefficient 1
beta_2	f4	1 s / semi-circle ²		model period coefficient 2
beta_3	f4	1 s / semi-circle ³		model period coefficient 3
Padding	u1[..]			Padding bytes, see 3.2.5

GPSUTC	Number: 5894
	"OnChange" interval: block generated each time subframe 4, page 18, is received from a GPS satellite

The GPSUTC block contains the decoded UTC data. These data are conveyed in subframes 4, page 18 of the satellite navigation message. Refer to GPS ICD for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the GPS satellite from which these UTC parameters have been received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
A_1	f4	1 s / s		first order term of polynomial
A_0	f8	1 s		constant term of polynomial
t_ot	u4	1 s		reference time for UTC data
WN_t	u1	1 week		UTC reference week number, to which t_ot is referenced
DEL_t_LS	i1	1 s		Delta time due to leap seconds whenever the effectivity time is not in the past

WN_LSF	u1	1 week		Effectivity time of leap second (week)
DN	u1	1 day		Effectivity time of leap second (day)
DEL_t_LSF	i1	1 s		Delta time due to leap seconds whenever the effectivity time is in the past
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.4 GLONASS Decoded Message Blocks

GLONav	Number: 4004
	"OnChange" interval: block generated each time a new navigation data set is received from a GLONASS satellite

The GLONav block contains the decoded ephemeris data for one GLONASS satellite.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
SVID	u1			ID of the GLONASS satellite for which ephemeris is provided in this block (see 3.2.9).
FreqNr	u1			Frequency number of the GLONASS satellite for which ephemeris is provided in this block (see 3.2.9).
X	f8	1000 m		x-component of satellite position in PZ-90.02
Y	f8	1000 m		y-component of satellite position in PZ-90.02
Z	f8	1000 m		z-component of satellite position in PZ-90.02
Dx	f4	1000 m / s		x-component of satellite velocity in PZ-90.02

Dy	f4	1000 m / s		y-component of satellite velocity in PZ-90.02
Dz	f4	1000 m / s		z-component of satellite velocity in PZ-90.02
Ddx	f4	1000 m / s ²		x-component of satellite acceleration in PZ-90.02
Ddy	f4	1000 m / s ²		y-component of satellite acceleration in PZ-90.02
Ddz	f4	1000 m / s ²		z-component of satellite acceleration in PZ-90.02
gamma	f4	1 Hz / Hz		$\gamma_n(t_b)$: relative deviation of predicted carrier frequency
tau	f4	1 s		$\tau_n(t_b)$: time correction to GLONASS time
dtau	f4	1 s		$\Delta\tau_n$: time difference between L2 and L1 sub-band
t_oe	u4	1 s		reference time-of-week in GPS time frame
WN_toe	u2	1 week		reference week number in GPS time frame (modulo 1024)
P1	u1	1 minute		time interval between adjacent values of t_b
P2	u1			1-bit odd/even flag of t_b
E	u1	1 day		age of data
B	u1			3-bit health flag, satellite unhealthy if MSB set
tb	u2	1 minute		time of day (center of validity interval)
M	u1			2-bit GLONASS-M satellite identifier (01, otherwise 00)
P	u1			2-bit mode of computation of time parameters
l	u1			1-bit health flag, 0=healthy, 1=unhealthy
P4	u1			1-bit 'updated' flag of ephemeris data
N_T	u2	1 day		current day number within 4-year interval
F_T	u2	0.01 m		predicted user range accuracy at time t_b
Padding	u1[..]			Padding bytes, see 3.2.5

GLOAlm	Number:	4005
	"OnChange" interval:	block generated each time a new almanac data set is received from a GLONASS satellite

The GLOAlm block contains the decoded navigation data for one GLONASS satellite.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
SVID	u1			ID of the GLONASS satellite for which almanac is provided in this block (see 3.2.9).
FreqNr	u1			Frequency number of the GLONASS satellite for which almanac is provided in this block (see 3.2.9). This number corresponds to the H_n^A parameter in the GLONASS ICD.
epsilon	f4			ϵ_n^A : orbit eccentricity
t_oa	u4	1 s		Reference time-of-week in GPS time frame
Delta_i	f4	1 semi-circle		Δi_n^A : correction to inclination
lambda	f4	1 semi-circle		λ_n^A : Longitude of first ascending node

t _{ln}	f4	1 s		$t_{\lambda_n}^A$: time of first ascending node passage
omega	f4	1 semi-circle		ω_n^A : argument of perigee
Delta_T	f4	1 s / orbit-period		ΔT_n^A : correction to mean Draconian period
dDelta_T	f4	1 s / orbit-period ²		$d\Delta T_n^A$: rate of change correction to mean Draconian period
tau	f4	1 s		τ_n^A : coarse correction to satellite time
WN _a	u1	1 week		Reference week in GPS time frame (modulo 256)
C	u1			C_n^A : 1-bit general health flag (1 indicates healthy)
N	u2	1 day		N^A : calendar day number within 4 year period
M	u1			M_n^A : 2-bit GLONASS-M satellite identifier
N ₄	u1			N_4 : 4 year interval number, starting from 1996
Padding	u1[..]			Padding bytes, see 3.2.5

GLoTime	Number: 4036 "OnChange" interval: block generated at the end of each GLONASS super-frame, i.e. every 2.5 minutes.
---------	--

The GLoTime block contains the decoded non-immediate data related to the difference between GLONASS and GPS, UTC and UT1 time scales.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			ID of the GLONASS satellite from which the data in this block has been decoded (see 3.2.9).
FreqNr	u1			Frequency number of the GLONASS satellite from which the data in this block has been decoded (see 3.2.9).
N_4	u1			4 year interval number, starting from 1996
KP	u1			notification of leap second
N	u2	1 day		calendar day number within 4 year period
tau_GPS	f4	1 s		difference with respect to GPS time
tau_c	f8	1 s		GLONASS time scale correction to UTC(SU)

B1	f4	1 s		difference between UT1 and UTC(SU)
B2	f4	1 s / msd		daily change of B1
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.5 Galileo Decoded Message Blocks

GALNav	Number:	4002
	"OnChange" interval:	output each time a new navigation data batch is decoded.

The `GalNav` block contains the following decoded navigation data for one Galileo satellite:

- orbital elements and clock corrections
- health, Signal-In-Space Accuracy (SISA) indexes and Broadcast Group Delays (BGDs) for each carrier or carrier combinations.

The interpretation of the clock correction parameters (`t_oc`, `a_f0`, `a_f1`, `a_f2`) depends on the value of the `Source` field:

Source	Message type	Applicable Clock Model
2	I/NAV	(L1,E5b)
16	F/NAV	(L1,E5a)

If the receiver is decoding both the I/NAV and the F/NAV data stream, it will output a `GalNav` block for the I/NAV stream, containing the (L1, E5b) clock model, and a different `GalNav` block for the F/NAV stream, containing the (L1, E5a) clock model.

Depending on the message type being decoded, some health, SISA or BGD values may not be available (in that case they are set to their respective Do-Not-Use values). The following health, SISA and BGD values are guaranteed to be available for a given value of the `Source` field:

Source	Health, and availability
2 (I/NAV)	At least L1-B _{DVS} , L1-B _{HS} , E5b _{DVS} , E5b _{HS} , SISA_L1E5b and BGD_L1E5b are available
16 (F/NAV)	At least E5a _{DVS} , E5a _{HS} , SISA_L1E5a and BGD_L1E5a are available

The `IODNav` field identifies the issue of data. All orbital elements, clock parameters and SISA values in the block are guaranteed to refer to the same data batch identified by `IODNav`. The fields `Health_OSSOL`, `BGD_L1E5a`, `BGD_L1E5b` and `CNAVenc` are not covered by the issue of data, and the block simply contains the latest received value.

Please refer to the Galileo Signal-In-Space ICD for the interpretation and usage of the parameters contained in this SBF block.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
<code>Sync1</code>	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
<code>Sync2</code>	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.

CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
SVID	u1			SVID of the Galileo satellite (see 3.2.9)
Source	u1			See table above: this field indicates how to interpret the clock correction parameters.
SQRT_A	f8	1 m ^{1/2}		Square root of the semi-major axis
M_0	f8	1 semi-circle		Mean anomaly at reference time
e	f8			Eccentricity
i_0	f8	1 semi-circle		Inclination angle at reference time
omega	f8	1 semi-circle		Argument of perigee
OMEGA_0	f8	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension
IDOT	f4	1 semi-circle / s		Rate of inclination angle
DEL_N	f4	1 semi-circle / s		Mean motion difference from computed value
C_uc	f4	1 rad		Amplitude of the cosine harmonic correction term to the argument of latitude
C_us	f4	1 rad		Amplitude of the sine harmonic correction term to the argument of latitude
C_rc	f4	1 m		Amplitude of the cosine harmonic correction term to the orbit radius
C_rs	f4	1 m		Amplitude of the sine harmonic correction term to the orbit radius
C_ic	f4	1 rad		Amplitude of the sine harmonic correction term to the angle of inclination
C_is	f4	1 rad		Amplitude of the cosine harmonic correction term to the angle of inclination
t_oe	u4	1 s		Reference time, ephemeris

t _{oc}	u4	1 s		Reference time, clock. The <i>Source</i> field indicates which clock model t _{oc} refers to.
a _{f2}	f4	1 s / s ²		SV clock aging. The <i>Source</i> field indicates which clock model a _{f2} refers to.
a _{f1}	f4	1 s / s		SV clock drift. The <i>Source</i> field indicates which clock model a _{f1} refers to.
a _{f0}	f8	1 s		SV clock bias. The <i>Source</i> field indicates which clock model a _{f0} refers to.
WNt _{oe}	u2	1 week		WN associated with t _{oe} , modulo 4096
WNt _{oc}	u2	1 week		WN associated with t _{oc} , modulo 4096
IODnav	u2			Issue of data, navigation (10 bits)
Health_OSSOL	u2			Bit field indicating the last received Health Status (HS) and Data Validity Status (DVS) of the E5a, E5b and L1-B signals: Bit 0: If set, bits 1 to 3 are valid, otherwise they must be ignored. Bit 1: 1-bit L1-B _{DVS} Bits 2-3: 2-bit L1-B _{HS} Bit 4: If set, bits 5 to 7 are valid, otherwise they must be ignored. Bit 5: 1-bit E5b _{DVS} Bits 6-7: 2-bit E5b _{HS} Bit 8: If set, bits 9 to 11 are valid, otherwise they must be ignored. Bit 9: 1-bit E5a _{DVS} Bits 10-11: 2-bit E5a _{HS} Bits 12-15: Reserved
Health_PRS	u1			Reserved
SISA_L1E5a	u1		255	Signal-In-Space Accuracy Index (L1, E5a)
SISA_L1E5b	u1		255	Signal-In-Space Accuracy Index (L1, E5b)
SISA_L1AE6A	u1		255	Reserved
BGD_L1E5a	f4	1 s	$-2 \cdot 10^{10}$	Last received broadcast group delay (L1, E5a)
BGD_L1E5b	f4	1 s	$-2 \cdot 10^{10}$	Last received broadcast group delay (L1, E5b)
BGD_L1AE6A	f4	1 s	$-2 \cdot 10^{10}$	Reserved
CNAVenc	u1		255	1-bit C/NAV encryption status from L1-B.
Padding	u1[.]			Padding bytes, see 3.2.5

GALAlm	Number:	4003
	"OnChange" interval:	output each time a new almanac set is received for a satellite.

The GalAlm block contains the decoded almanac data for one Galileo satellite.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
SVID	u1			SVID of the Galileo satellite from which these almanac parameters have been received (see 3.2.9)
Source	u1			See corresponding field in the GalNav block. Source can take the value 18 to indicate that the almanac data contained in this block has been merged from INAV and FNAV pages.
e	f4			Eccentricity
t_oa	u4	1 s		almanac reference time of week
delta_i	f4	1 semi-circle		Inclination angle at reference time, relative to nominal
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension

SQRT_A	f4	1 m ^{1/2}		Square root of the semi-major axis, relative to nominal
OMEGA_0	f4	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
omega	f4	1 semi-circle		Argument of perigee
M_0	f4	1 semi-circle		Mean anomaly at reference time
a_f1	f4	1 s / s		SV clock drift
a_f0	f4	1 s		SV clock bias
WN_a	u1	1 week		2-bit almanac reference week
SVID_A	u1			SVID of the Galileo satellite of which the almanac parameters are provided in this block (see 3.2.9 for the SVID numbering convention).
health	u2			<p>Bit field indicating the health status (HS) of the E5a, E5b, L1-B, L1-A and E6-A signals:</p> <p>Bit 0: If set, bits 1 and 2 are valid, otherwise they must be ignored.</p> <p>Bits 1-2: 2-bit L1-B_{HS}</p> <p>Bit 3: If set, bits 4 and 5 are valid, otherwise they must be ignored.</p> <p>Bits 4-5: 2-bit E5b_{HS}</p> <p>Bit 6: If set, bits 7 and 8 are valid, otherwise they must be ignored.</p> <p>Bits 7-8: 2-bit E5a_{HS}</p> <p>Bit 9: If set, bits 10 and 11 are valid, otherwise they must be ignored.</p> <p>Bits 10-11: 2-bit L1-A_{HS}</p> <p>Bit 12: If set, bits 13 and 14 are valid, otherwise they must be ignored.</p> <p>Bits 13-14: 2-bit E6-A_{HS}</p> <p>Bit 15: Reserved</p>
IODa	u1			4-bit Issue of Data for the almanac.
Padding	u1[..]			Padding bytes, see 3.2.5

GALION	Number:	4030
	"OnChange" interval:	output each time the ionospheric parameters are received from a Galileo satellite.

The `GALION` block contains the decoded ionosphere model parameters of the Galileo system.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNC	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNC</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNC</code> is also the Galileo week number plus 1024.

SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 3.2.9)
Source	u1			Message type from which the data has been decoded: 2: I/NAV 16: F/NAV
a_i0	f4	$1 \cdot 10^{-22} \text{ W / (m}^2 \text{ Hz)}$		Effective ionization level, a _{i0}
a_i1	f4	$1 \cdot 10^{-22} \text{ W / (m}^2 \text{ Hz) / deg}$		Effective ionization level, a _{i1}
a_i2	f4	$1 \cdot 10^{-22} \text{ W / (m}^2 \text{ Hz) / deg}^2$		Effective ionization level, a _{i2}
StormFlags	u1			Bit field containing the five ionospheric storm flags: Bit 0: SF5 Bit 1: SF4 Bit 2: SF3 Bit 3: SF2 Bit 4: SF1 Bits 5-7: Reserved
Padding	u1[..]			Padding bytes, see 3.2.5

GALUTC	Number:	4031
	"OnChange" interval:	output each time the UTC offset parameters are received from a Galileo satellite.

The GalUTC block contains the decoded UTC parameter information.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 3.2.9)
Source	u1			Message type from which the data has been decoded: 2: I/NAV 16: F/NAV
A_1	f4	1 s / s		first order term of polynomial
A_0	f8	1 s		constant term of polynomial
t_ot	u4	1 s		reference time of week for UTC data
WN_ot	u1	1 week		UTC reference week number, to which t_ot is referenced

DEL_t_LS	i1	1 s		Delta time due to leap seconds whenever the effectivity time is not in the past
WN_LSF	u1	1 week		Effectivity time of leap second (week)
DN	u1	1 day		Effectivity time of leap second (day)
DEL_t_LSF	i1	1 s		Delta time due to leap seconds whenever the effectivity time is in the past
Padding	u1[..]			Padding bytes, see 3.2.5

GALGstGps	Number:	4032
	"OnChange" interval:	output each time valid GST-GPS offset parameters are received from a Galileo satellite.

This block contains the decoded GPS to Galileo System Time offset parameters. This block is only output if these parameters are valid in the navigation page (i.e. if they are not set to "all ones").

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 3.2.9)
Source	u1			Message type from which the data has been decoded: 2: I/NAV 16: F/NAV
A_1G	f4	1 s / s		Rate of change of the offset
A_0G	f4	1 s		Constant term of the offset
t_oG	u4	1 s		Reference time of week

WN_oG	u1	1 week		6-bit reference week number.
Padding	u1[..]			Padding bytes, see 3.2.5

GALSARRLM	Number: 4034
	"OnChange" interval: generated each time a SAR RLM message is decoded.

This block contains a decoded Galileo search-and-rescue (SAR) return link message (RLM).

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SVID	u1			SVID of the Galileo satellite from which this RLM has been received.
Source	u1			Message type from which the data has been decoded: 2: I/NAV 16: F/NAV
RLMLength	u1			Length of the RLM message in bits. <code>RLMLength</code> can be either 80 for a short message or 160 for a long message.
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software

RLMbits	u4[N]			<p>Bits in the RLM message, with the first bit being the MSB of RLMbits[0].</p> <p><i>N</i> is 3 for a short message (i.e. if RLMLength is 80), and 5 for a long message (i.e. if RLMLength is 160).</p> <p>The 16 unused bits of a short message are set to 0. These bits correspond to the 16 LSBs of RLMbits[2].</p>
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.6 Compass/BeiDou Decoded Message Blocks

CMPNav	Number: 4081 "OnChange" interval: block generated each time a new navigation data set is received from a Compass/BeiDou satellite
--------	--

The **CMPNav** block contains the decoded navigation data for one Compass/BeiDou satellite. The navigation data is received from the B1I signal modulated by either the D1 or D2 message.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW . WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the Compass/BeiDou satellite of which the ephemeris is given in this block (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
WN	u2	1 week		BeiDou week number as received from the navigation message (from 0 to 8191)
URA	u1			User range accuracy index (4-bit value)

SatH1	u1			1-bit autonomous health
IODC	u1			Issue of data, clock (5 bits)
IODE	u1			Issue of data, ephemeris (5 bits)
Reserved2	u2			unused, to be ignored by decoding software
T_GD1	f4	1 s		B1 equipment group delay differential
T_GD2	f4	1 s	$-2 \cdot 10^{10}$	B2 equipment group delay differential (set to the do-not-use value when unknown)
t_oc	u4	1 s		clock data reference time, in BeiDou system time (lagging GPS time by 14 seconds).
a_f2	f4	1 s / s ²		SV clock aging
a_f1	f4	1 s / s		SV clock drift
a_f0	f4	1 s		SV clock bias
C_rs	f4	1 m		Amplitude of the sine harmonic correction term to the orbit radius
DEL_N	f4	1 semi-circle / s		Mean motion difference from computed value
M_0	f8	1 semi-circle		Mean anomaly at reference time
C_uc	f4	1 rad		Amplitude of the cosine harmonic correction term to the argument of latitude
e	f8			Eccentricity
C_us	f4	1 rad		Amplitude of the sine harmonic correction term to the argument of latitude
SQRT_A	f8	1 m ^{1/2}		Square root of the semi-major axis
t_oe	u4	1 s		Reference time ephemeris, in BeiDou system time (lagging GPS time by 14 seconds).
C_ic	f4	1 rad		Amplitude of the cosine harmonic correction term to the angle of inclination
OMEGA_0	f8	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
C_is	f4	1 rad		Amplitude of the sine harmonic correction term to the angle of inclination
i_0	f8	1 semi-circle		Inclination angle at reference time
C_rc	f4	1 m		Amplitude of the cosine harmonic correction term to the orbit radius
omega	f8	1 semi-circle		Argument of perigee
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension
IDOT	f4	1 semi-circle / s		Rate of inclination angle
WNt_oc	u2	1 week		BeiDou week number associated with t_oc, modulo 8192. Note that this value relates to the BeiDou system time.
WNt_oe	u2	1 week		BeiDou week number associated with t_oe, modulo 8192. Note that this values relates to the BeiDou system time.
Padding	u1[.]			Padding bytes, see 3.2.5

3.3.7 QZSS Decoded Message Blocks

QZSNav	Number: 4095
	"OnChange" interval: block generated each time a new navigation data set is received from a QZSS satellite

The **QZSNav** block contains the decoded navigation data for one QZSS satellite. The data is decoded from the navigation message transmitted in the L1 C/A signal. Refer to the QZSS ICD for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW . WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the QZSS satellite of which the ephemeris is given in this block (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
WN	u2	1 week	65535	Week number (10 bits from subframe 1, word 3)
CAorPonL2	u1			Code(s) on L2 channel (2 bits from subframe 1, word 3). Always 2 for QZSS satellites.

URA	u1			User Range accuracy index (4 bits from subframe 1 word 3)
health	u1			6-bit health from subframe 1, word 3 (6 bits from subframe 1, word 3)
L2DataFlag	u1			Data flag for L2 P-code (1 bit from subframe 1, word 4). Always 1 for QZSS satellites.
IODC	u2			Issue of data, clock (10 bits from subframe 1)
IODE2	u1			Issue of data, ephemeris (8 bits from subframe 2)
IODE3	u1			Issue of data, ephemeris (8 bits from subframe 3)
FitIntFlg	u1			Curve Fit Interval, (1 bit from subframe 2, word 10)
Reserved2	u1			unused, to be ignored by decoding software
T_gd	f4	1 s	$-2 \cdot 10^{10}$	Estimated group delay differential
t_oc	u4	1 s		clock data reference time
a_f2	f4	1 s / s ²		SV clock aging
a_f1	f4	1 s / s		SV clock drift
a_f0	f4	1 s		SV clock bias
C_rs	f4	1 m		Amplitude of the sine harmonic correction term to the orbit radius
DEL_N	f4	1 semi-circle / s		Mean motion difference from computed value
M_0	f8	1 semi-circle		Mean anomaly at reference time
C_uc	f4	1 rad		Amplitude of the cosine harmonic correction term to the argument of latitude
e	f8			Eccentricity
C_us	f4	1 rad		Amplitude of the sine harmonic correction term to the argument of latitude
SQRT_A	f8	1 m ^{1/2}		Square root of the semi-major axis
t_oe	u4	1 s		Reference time ephemeris
C_ic	f4	1 rad		Amplitude of the cosine harmonic correction term to the angle of inclination
OMEGA_0	f8	1 semi-circle		Longitude of ascending node of orbit plane at weekly epoch
C_is	f4	1 rad		Amplitude of the sine harmonic correction term to the angle of inclination
i_0	f8	1 semi-circle		Inclination angle at reference time
C_rc	f4	1 m		Amplitude of the cosine harmonic correction term to the orbit radius
omega	f8	1 semi-circle		Argument of perigee
OMEGADOT	f4	1 semi-circle / s		Rate of right ascension
IDOT	f4	1 semi-circle / s		Rate of inclination angle
WNt_oc	u2	1 week		WN associated with t_oc, modulo 1024
WNt_oe	u2	1 week		WN associated with t_oe, modulo 1024
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.8 SBAS Decoded Message Blocks

In the SBAS message blocks described in the next pages, the time tag reported in the `TOW` and `WNC` fields always refers to the end of the last bit of the message. To get the time of transmission of the beginning of the first bit of the message, which is equal to the time of applicability of the SBAS navigation data, the user must subtract 1 second from `TOW`.

The receiver is receiving SBAS data from all the tracked SBAS satellites, but decoding of the messages is performed only from the L1 signal of the satellite that is currently used to compute corrections. Therefore all the SBF blocks in the next pages are available only for this satellite.

Note that a user interested in the actual SBAS corrections that have been applied in the position computation can also use the `GEOCorrections` block.

GEOMT00	Number: 5925
	"OnChange" interval: block generated each time an empty MT00 is received from an SBAS satellite

This block is sent to indicate that an empty SBAS message type 0 has been received.

Depending on the SBAS operational mode, message type 0 can contain the contents of message type 2. Upon reception of a message type 0, the receiver checks whether the message is empty (it contains only 0's) or whether it contains the message type 2 contents. In the former case, a GEOMT00 block will be generated. In the latter case, a GEOFastCorr block will be generated. Refer to section A.4.4.1 of the DO 229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
Padding	u1[...]			Padding bytes, see 3.2.5

GEOPRNMask	Number: 5926 "OnChange" interval: block generated each time MT01 is received from an SBAS satellite
------------	--

This block contains the decoded PRN mask transmitted in SBAS message type 1. Refer to section A.4.4.2 of the DO 229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
IODP	u1			Issue of data - PRN.
NbrPRNs	u1			Number of PRNs designated in the mask.

PRNMask	u1[NbrPRNs]			<p>List of the PRNs in the PRN mask.</p> <p>PRNMask[0] is the first PRN designated in the PRN mask (from 1 to 210), PRNMask[1] is the 2nd PRN designated in the PRN mask, etc...</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GEOFastCorr	<p>Number: 5927</p> <p>"OnChange" interval: block generated each time MT02, MT03, MT04, MT05, MT24 and possibly MT00 is received from an SBAS satellite</p>
-------------	---

This block contains the decoded fast corrections transmitted in the SBAS message types 2, 3, 4, 5, 24 and possibly 0 if the type 0 message contains the type 2 contents. Refer to section A.4.4.3 and A.4.4.8 of the DO 229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description								
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.								
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.								
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.								
ID	u2		ID	The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)								
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.								
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.								
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.								
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)								
MT	u1			Message type from which these fast corrections come, either 0, 2, 3, 4, 5 or 24.								
IODP	u1			Issue of data - PRN.								
IODF	u1			Issue of data - fast corrections.								
N	u1			Number of fast correction sets in this message. This is the number of <code>FastCorr</code> sub-blocks. <code>N</code> depends on the message type as follows. <div><table><tr><th>Message type</th><th>N</th></tr><tr><td>MT00, MT02, MT03, MT04</td><td>13</td></tr><tr><td>MT05</td><td>12</td></tr><tr><td>MT24</td><td>6</td></tr></table></div>	Message type	N	MT00, MT02, MT03, MT04	13	MT05	12	MT24	6
Message type	N											
MT00, MT02, MT03, MT04	13											
MT05	12											
MT24	6											
SBLength	u1			Length of the <code>FastCorr</code> sub-blocks in bytes								
<i>FastCorr</i>		<i>A succession of N FastCorr sub-blocks, see definition below</i>								

Padding	u1[..]			Padding bytes, see 3.2.5
---------	--------	--	--	--------------------------

FastCorr sub-block definition:

Parameter	Type	Units & Scale Factor	Description
PRNMaskNo	u1		Sequence number in the PRN mask, from 1 to 51.
UDREI	u1		User Differential Range Error Indicator for the PRN at index PRNMaskNo.
Reserved	u1[2]		Reserved for future use, to be ignored by decoding software
PRC	f4	1 m	Pseudorange correction for the PRN at index PRNMaskNo.
Padding	u1[..]		Padding bytes, see 3.2.5

GEOIntegrity	Number:	5928
	"OnChange" interval:	block generated each time MT06 is received from an SBAS satellite

This block contains the decoded integrity information transmitted in SBAS message type 6. Refer to section A.4.4.4 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
IODF	u1[4]			Issue of data - fast corrections for MT02, MT03, MT04 and MT05.
UDREI	u1[51]			User Differential Range Error Indicator for each of the 51 slots in the PRN mask.
Padding	u1[..]			Padding bytes, see 3.2.5

GEOFastCorrDegr	Number: 5929
	"OnChange" interval: block generated each time MT07 is received from an SBAS satellite

This block contains the decoded fast correction degradation factors transmitted in SBAS message type 7. Refer to section A.4.4.5 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
IODP	u1			Issue of data - PRN.
t_lat	u1	1 s		System latency.
ai	u1[51]			Degradation factor indicator (from 0 to 15) for each of the 51 slots in the PRN mask.
Padding	u1[..]			Padding bytes, see 3.2.5

GEONav	Number: 5896
	"OnChange" interval: block generated each time MT09 is received from an SBAS satellite

This block contains the decoded navigation data transmitted in SBAS message type 9. Refer to section A.4.4.11 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite of which the navigation data is provided here (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
IODN	u2			Issue of data - navigation (DO 229-B) Spare (DO 229-C)
URA	u2			Accuracy exponent
t0	u4	1 s		Time of applicability (time-of-day)
Xg	f8	1 m		X position at time-of-day t0
Yg	f8	1 m		Y position at time-of-day t0

Zg	f8	1 m		Z position at time-of-day t_0
Xgd	f8	1 m / s		X velocity at time-of-day t_0
Ygd	f8	1 m / s		Y velocity at time-of-day t_0
Zgd	f8	1 m / s		Z velocity at time-of-day t_0
Xgdd	f8	1 m / s ²		X acceleration at time-of-day t_0
Ygdd	f8	1 m / s ²		Y acceleration at time-of-day t_0
Zgdd	f8	1 m / s ²		Z acceleration at time-of-day t_0
aGf0	f4	1 s		Time offset with respect to SBAS network time
aGf1	f4	1 s / s		Time drift with respect to SBAS network time
Padding	u1[..]			Padding bytes, see 3.2.5

GEODegrFactors	Number: 5930
	"OnChange" interval: block generated each time MT10 is received from an SBAS satellite

This block contains the decoded degradation factors transmitted in SBAS message type 10. Refer to section A.4.5 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
Brrc	f8	1 m		A parameter associated with the relative estimation noise and round-off error.
Cltc_lsb	f8	1 m		Maximum round-off error due to the LSB resolution of the orbit and clock information.
Cltc_v1	f8	1 m / s		Velocity error bound on the maximum range rate difference of missed messages due to clock and orbit rate differences.

Iltc_v1	u4	1 s		Update interval for long term corrections when the velocity code is 1.
Cltc_v0	f8	1 m		Bound on the update delta between successive long term corrections.
Iltc_v0	u4	1 s		Minimum update interval for long term messages when the velocity code is 0.
Cgeo_lsb	f8	1 m		Maximum round-off error due to the LSB resolution of the orbit and clock information.
Cgeo_v	f8	1 m / s		Velocity error bound on the maximum range rate difference of missed messages due to clock and orbit rate differences.
Igeo	u4	1 s		Update interval for GEO navigation messages.
Cer	f4	1 m		A degradation parameter.
Ciono_step	f8	1 m		Bound on the difference between successive ionospheric grid delay values.
Iiono	u4	1 s		Minimum update interval for ionospheric correction messages.
Ciono_ramp	f8	1 m / s		Rate of change of the ionospheric corrections.
RSSudre	u1			Root-sum-square flag (UDRE)
RSSiono	u1			Root-sum-square flag (IONO)
Reserved2	u1[2]			Reserved for future use, to be ignored by decoding software
Ccovariance	f8			A parameter used to compensate for the errors introduced by quantization (introduced in DO 229-C). To be multiplied by the SF parameter from the GEOClockEphCovMatrix block.
Padding	u1[..]			Padding bytes, see 3.2.5

GEONetworkTime	Number: 5918 "OnChange" interval: block generated each time MT12 is received from an SBAS satellite
----------------	--

This block contains the decoded network time offset parameters transmitted in SBAS message type 12. Refer to section A.4.4.15 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which this Network Time data was received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
A_1	f4	1 s / s		first order term of polynomial
A_0	f8	1 s		constant term of polynomial
t_ot	u4	1 s		reference time for UTC data (time of week)
WN_t	u1	1 week		UTC reference week number, to which <code>t_ot</code> is referenced
DEL_t_LS	i1	1 s		Delta time due to leap seconds whenever the effectivity time is not in the past

WN_LSF	u1	1 week		Effectivity time of leap second (week)
DN	u1	1 day		Effectivity time of leap second (day)
DEL_t_LSF	i1	1 s		Delta time due to leap seconds whenever the effectivity time is in the past
UTC_std	u1			UTC Standard Identifier
GPS_WN	u2	1 week		GPS week number (modulo 1024)
GPS_TOW	u4	1 s		GPS time-of-week
GlonassID	u1			Glonass Indicator
Padding	u1[..]			Padding bytes, see 3.2.5

GEOAlm	Number: 5897 "OnChange" interval: block generated each time MT17 is received from an SBAS satellite
--------	--

This block contains the decoded almanac data for one SBAS satellite, as transmitted in SBAS message type 17. A different GEOAlm block is generated for each of the up to three almanac data sets in MT17. Refer to section A.4.4.12 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite of which the almanac is provided here (see 3.2.9)
Reserved0	u1			Reserved for future use, to be ignored by decoding software
DataID	u1			Data ID
Reserved1	u1			Reserved for future use, to be ignored by decoding software
Health	u2			Health bits
t_oa	u4	1 s		Time of applicability (time-of-day)

Xg	f8	1 m		X position at time-of-day t_0
Yg	f8	1 m		Y position at time-of-day t_0
Zg	f8	1 m		Z position at time-of-day t_0
Xgd	f8	1 m / s		X velocity at time-of-day t_0
Ygd	f8	1 m / s		Y velocity at time-of-day t_0
Zgd	f8	1 m / s		Z velocity at time-of-day t_0
Padding	u1[..]			Padding bytes, see 3.2.5

GEOIGPMask	Number: 5931 "OnChange" interval: block generated each time MT18 is received from an SBAS satellite
------------	--

This block contains the decoded ionospheric grid point mask transmitted in SBAS message type 18. Refer to section A.4.4.9 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
NbrBands	u1			Number of bands being broadcast.
BandNbr	u1			Band number.
IODI	u1			Issue of data - ionosphere.
NbrIGPs	u1			Number of ionospheric grid points (IGP) designated in the mask.

IGPMask	u1[NbrIGPs]			<p>List of the IGPs in the IGP mask.</p> <p>IGPMask[0] is the first IGP designated in the IGP mask (from 1 to 201), IGPMask[1] is the 2nd IGP designated in the IGP mask, etc...</p>
Padding	u1[..]			Padding bytes, see 3.2.5

GEOLongTermCorr	Number: 5932 "OnChange" interval: block generated each time MT24 or MT25 is received from an SBAS satellite
-----------------	--

This block contains the decoded long term corrections transmitted in SBAS message types 24 and 25. Refer to section A.4.4.7 and A.4.4.8 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
N	u1			Number of long-term corrections in this message. This is the number of <code>LTCorr</code> sub-blocks. <code>N</code> can be 0, 1, 2, 3 or 4.
SBLength	u1	1 byte		Length of the <code>LTCorr</code> sub-blocks in bytes
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
LTCorr		A succession of <code>N</code> <code>LTCorr</code> sub-blocks, see definition below

Padding	u1[..]		Padding bytes, see 3.2.5
---------	--------	--	--------------------------

LTCorr sub-block definition:

Parameter	Type	Units & Scale Factor	Description
VelocityCode	u1		Velocity code (0 or 1)
PRNMaskNo	u1		Sequence in the PRN mask, from 1 to 51. Note that if the PRN mask No. from the original message is 0, the corresponding long term corrections are ignored, and hence not included in the GEOLongTermCorr block.
IODP	u1		Issue of data - PRN.
IODE	u1		Issue of data - ephemeris.
dx	f4	1 m	Satellite position offset (x).
dy	f4	1 m	Satellite position offset (y).
dz	f4	1 m	Satellite position offset (z).
dxRate	f4	1 m / s	Satellite velocity offset (x), or 0.0 if VelocityCode is 0.
dyRate	f4	1 m / s	Satellite velocity offset (y), or 0.0 if VelocityCode is 0.
dzRate	f4	1 m / s	Satellite velocity offset (z), or 0.0 if VelocityCode is 0.
da_f0	f4	1 s	Satellite clock offset.
da_f1	f4	1 s / s	Satellite drift correction, or 0.0 if VelocityCode is 0.
t_oe	u4	1 s	Time-of-day of applicability, or 0 if VelocityCode is 0.
Padding	u1[..]		Padding bytes, see 3.2.5

GEOIonoDelay	Number: 5933
	"OnChange" interval: block generated each time MT26 is received from an SBAS satellite

This block contains the decoded ionospheric delays transmitted in SBAS message type 26. Refer to section A.4.4.10 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which the message has been received (see 3.2.9)
BandNbr	u1			Band number
IODI	u1			Issue of data - ionosphere.
N	u1			Number of ionospheric delay corrections in this message. This is the number of <code>IDC</code> sub-blocks. <code>N</code> is always 15.
SBLength	u1	1 byte		Length of the <code>IDC</code> sub-blocks in bytes.
Reserved	u1			Reserved for future use, to be ignored by decoding software

<i>IDC</i>		<i>A succession of N <i>IDC</i> sub-blocks, see definition below</i>
Padding	u1[..]			Padding bytes, see 3.2.5

IDC sub-block definition:

Parameter	Type	Units & Scale Factor	Description
IGPMaskNo	u1		Sequence number in the IGP mask (see <i>GEOIGPMask</i> block), from 1 to 201.
GIVEI	u1		Grid Ionospheric Vertical Error Indicator, from 0 to 15
Reserved	u1[2]		Reserved for future use, to be ignored by decoding software
VerticalDelay	f4	1 m	IGP vertical delay estimate.
Padding	u1[..]		Padding bytes, see 3.2.5

GEOServiceLevel	<p>Number: 5917</p> <p>"OnChange" interval: block generated each time MT27 is received from an SBAS satellite</p>
-----------------	---

This block contains a decoded service level message for a geostationary SBAS satellite as sent in message type 27. Refer to section A.4.4.13 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			ID of the SBAS satellite from which this service level message was received (see 3.2.9)
Reserved	u1			Reserved for future use, to be ignored by decoding software
IODS	u1			Issue of Data Service level, ranging from 0 to 7
nrMessages	u1			Number of service messages (MT27), from 1 to 8
MessageNR	u1			Service message number, from 1 to 8
PriorityCode	u1			Priority Code, from 0 to 3
dUDREI_In	u1			δ UDRE Indicator for users inside the service region, from 0 to 15
dUDREI_Out	u1			δ UDRE Indicator for users outside the service region, from 0 to 15
N	u1			Number of Regions in this message. This is the number of <code>ServiceRegion</code> sub-blocks. Ranging from 0 to 7
SBLength	u1	1 byte		Length of the <code>ServiceRegion</code> sub-blocks in bytes

<i>Regions</i>		<i>A succession of N ServiceRegion sub-blocks, see definition below</i>
Padding	u1[..]			Padding bytes, see 3.2.5

ServiceRegion sub-block definition:

Parameter	Type	Units & Scale Factor	Description
Latitude1	i1	1 degree	Coordinate 1 latitude, from -90 to +90
Latitude2	i1	1 degree	Coordinate 2 latitude, from -90 to +90
Longitude1	i2	1 degree	Coordinate 1 longitude, from -180 to +180
Longitude2	i2	1 degree	Coordinate 2 longitude, from -180 to +180
RegionShape	u1		Region Shape: 0=triangular, 1=square
Padding	u1[..]		Padding bytes, see 3.2.5

GEOClockEphCovMatrix	Number:	5934
	"OnChange" interval:	block generated each time MT28 is received from an SBAS satellite

This block contains the decoded clock-ephemeris covariance Cholesky factor matrix transmitted in SBAS message type 28. Refer to section A.4.4.16 of the DO-229 standard for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
PRN	u1			
IODP	u1			Issue of data - PRN.
N	u1			Number of covariance matrices in this message. This is the number of <code>CovMatrix</code> sub-blocks. <code>N</code> can be 1 or 2.
SBLength	u1	1 byte		Length of the <code>CovMatrix</code> sub-blocks in bytes
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software
CovMatrix		A succession of <code>N</code> <code>CovMatrix</code> sub-blocks, see definition below

Padding	u1[..]		Padding bytes, see 3.2.5
---------	--------	--	--------------------------

CovMatrix sub-block definition:

Parameter	Type	Units & Scale Factor	Description
PRNMaskNo	u1		Sequence number in the PRN mask, from 1 to 51. Note that if the PRN mask No. from the original message is 0, the corresponding matrix is ignored, and hence not included in the GEOClockEphCovMatrix block.
Reserved	u1[2]		Reserved for future use, to be ignored by decoding software
ScaleExp	u1		Scale exponent; scale factor (= $2^{(\text{scale exponent} - 5)}$)
E11	u2		$E_{1,1}$
E22	u2		$E_{2,2}$
E33	u2		$E_{3,3}$
E44	u2		$E_{4,4}$
E12	i2		$E_{1,2}$
E13	i2		$E_{1,3}$
E14	i2		$E_{1,4}$
E23	i2		$E_{2,3}$
E24	i2		$E_{2,4}$
E34	i2		$E_{3,4}$
Padding	u1[..]		Padding bytes, see 3.2.5

3.3.9 Position, Velocity and Time Blocks

PVTCartesian	Number: 4006
	"OnChange" interval: 10 ms

This block contains the position, velocity and time (PVT) solution at the time specified in the `TOW` and `WNc` fields. The time of applicability is specified in the receiver time frame.

The computed position (x , y , z) and velocity (v_x , v_y , v_z) are reported in a Cartesian coordinate system using the datum indicated in the `Datum` field. The position is that of the marker. The ARP-to-marker offset is set through the command **setAntennaOffset**.

The PVT solution is also available in ellipsoidal form in the `PVTGeodetic` block.

The variance-covariance information associated with the reported PVT solution can be found in the `PosCovCartesian` and `VelCovCartesian` blocks.

If no PVT solution is available, the `Error` field indicates the cause of the unavailability and all fields after the `Error` field are set to their respective Do-Not-Use values.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
X	f8	1 m	$-2 \cdot 10^{10}$	Marker X coordinate in coordinate frame specified by <code>Datum</code>
Y	f8	1 m	$-2 \cdot 10^{10}$	Marker Y coordinate in coordinate frame specified by <code>Datum</code>
Z	f8	1 m	$-2 \cdot 10^{10}$	Marker Z coordinate in coordinate frame specified by <code>Datum</code>
Undulation	f4	1 m	$-2 \cdot 10^{10}$	Geoid undulation computed from the global geoid model defined in the document 'Technical Characteristics of the NAVSTAR GPS, NATO, June 1991'
Vx	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the X direction
Vy	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the Y direction
Vz	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the Z direction

COG	f4	1 degree	$-2 \cdot 10^{10}$	Course over ground: this is defined as the angle of the vehicle with respect to the local level North, ranging from 0 to 360, and increasing towards east. Set to the do-not-use value when the speed is lower than 0.1m/s.
RxClkBias	f8	1 ms	$-2 \cdot 10^{10}$	Receiver clock bias relative to system time reported in the <code>TimeSystem</code> field. To transfer the receiver time to the system time, use: $t_{GPS/GST} = t_{rx} - RxClkBias$
RxClkDrift	f4	1 ppm	$-2 \cdot 10^{10}$	Receiver clock drift relative to system time (relative frequency error)
TimeSystem	u1		255	Time system of which the offset is provided in this sub-block: 0: GPS time 1: Galileo time 3: GLONASS time
Datum	u1		255	This field defines in which datum the coordinates are expressed: 0: WGS84/ITRS 19: Datum equal to that used by the DGNSS/RTK base station 30: ETRS89 (ETRF2000 realization) 31: NAD83(2011), North American Datum (2011) 32: NAD83(PA11), North American Datum, Pacific plate (2011) 33: NAD83(MA11), North American Datum, Marianas plate (2011) 34: GDA94(2010), Geocentric Datum of Australia (2010) 250: First user-defined datum 251: Second user-defined datum
NrSV	u1		255	Total number of satellites used in the PVT computation.
WACorrInfo	u1		0	Bit field providing information about which wide area corrections have been applied: Bit 0: set if orbit and satellite clock correction information is used Bit 1: set if range correction information is used Bit 2: set if ionospheric information is used Bit 3: set if orbit accuracy information is used (UERE/SISA) Bit 4: set if DO229 Precision Approach mode is active Bits 5-7: Reserved
ReferenceID	u2		65535	This field indicates the reference ID of the differential information used. In case of DGPS or RTK operation, this field is to be interpreted as the base station identifier. In SBAS operation, this field is to be interpreted as the PRN of the geostationary satellite used (from 120 to 158). If multiple base stations or multiple geostationary satellites are used the value is set to 65534.
MeanCorrAge	u2	0.01 s	65535	In case of DGPS or RTK, this field is the mean age of the differential corrections. In case of SBAS operation, this field is the mean age of the 'fast corrections' provided by the SBAS satellites.

SignalInfo	u4		0	Bit field indicating the type of GNSS signals having been used in the PVT computations. If a bit i is set, the signal type having index i has been used. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of <code>SignalInfo</code> .
AlertFlag	u1		0	<p>Bit field indicating integrity related information:</p> <p>Bits 0-1: RAIM integrity flag:</p> <ul style="list-style-type: none"> 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved <p>Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm)</p> <p>Bit 3: Reserved</p> <p>Bit 4: set if either the horizontal or the vertical 2DRMS accuracy is higher than the horizontal or vertical alert limits set by the <code>setNWALevels</code> command.</p> <p>Bits 5-7: Reserved</p>
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	<p>Bit field containing PPP-related information:</p> <p>Bits 0-11: Age of the last seed, in seconds. The age is clipped to 4091s. This field must be ignored when the seed type is 0 (see bits 13-15 below).</p> <p>Bit 12: Reserved</p> <p>Bits 13-15: Type of last seed:</p> <ul style="list-style-type: none"> 0: Not seeded or not in PPP positioning mode 1: Manual seed 2: Seeded from DGPS 3: Seeded from RTKFixed
Latency	u2	0.0001 s	65535	Reserved for future use
HAccuracy	u2	0.01 m	65535	2DRMS horizontal accuracy: twice the root-mean-square of the horizontal distance error. The horizontal distance between the true position and the computed position is expected to be lower than <code>HAccuracy</code> with a probability of at least 95%. The value is clipped to 65534 = 655.34m
VAccuracy	u2	0.01 m	65535	2DRMS vertical accuracy: twice the root-mean-square of the vertical error. The vertical distance between the true position and the computed position is expected to be lower than <code>VAccuracy</code> with a probability of at least 95%. The value is clipped to 65534 = 655.34m.

Rev 1

Rev 2

Rev 2

Misc	u1			<p>Bit field containing miscellaneous flags:</p> <p>Bit 0: In DGNSS or RTK mode, set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown.</p> <p>Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown.</p> <p>Bit 2: Proprietary.</p> <p>Bit 3: Proprietary.</p> <p>Bits 4-7: Reserved</p>
Padding	u1[..]			Padding bytes, see 3.2.5

PVTGeodetic	Number: 4007 "OnChange" interval: 10 ms
-------------	--

This block contains the position, velocity and time (PVT) solution at the time specified in the `TOW` and `WNc` fields. The time of applicability is specified in the receiver time frame.

The computed position (ϕ, λ, h) and velocity (v_n, v_e, v_u) are reported in an ellipsoidal coordinate system using the datum indicated in the `Datum` field. The velocity vector is expressed relative to the local-level Cartesian coordinate frame with north-, east-, up-unit vectors. The position is that of the marker. The ARP-to-marker offset is set through the command `setAntennaOffset`.

The PVT solution is also available in Cartesian form in the `PVTCartesian` block.

The variance-covariance information associated with the reported PVT solution can be found in the `PosCovGeodetic` and `VelCovGeodetic` blocks.

If no PVT solution is available, the `Error` field indicates the cause of the unavailability and all fields after the `Error` field are set to their respective Do-Not-Use values.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Latitude	f8	1 rad	$-2 \cdot 10^{10}$	Marker latitude, from $-\pi/2$ to $+\pi/2$, positive North of Equator
Longitude	f8	1 rad	$-2 \cdot 10^{10}$	Marker longitude, from $-\pi$ to $+\pi$, positive East of Greenwich
Height	f8	1 m	$-2 \cdot 10^{10}$	Marker ellipsoidal height (with respect to the ellipsoid specified by <code>Datum</code>)
Undulation	f4	1 m	$-2 \cdot 10^{10}$	Geoid undulation computed from the global geoid model defined in the document 'Technical Characteristics of the NAVSTAR GPS, NATO, June 1991'
Vn	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the North direction
Ve	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the East direction
Vu	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the 'Up' direction

COG	f4	1 degree	$-2 \cdot 10^{10}$	Course over ground: this is defined as the angle of the vehicle with respect to the local level North, ranging from 0 to 360, and increasing towards east. Set to the do-not-use value when the speed is lower than 0.1m/s.
RxClkBias	f8	1 ms	$-2 \cdot 10^{10}$	Receiver clock bias relative to system time reported in the <code>TimeSystem</code> field. To transfer the receiver time to the system time, use: $t_{GPS/GST} = t_{rx} - RxClkBias$
RxClkDrift	f4	1 ppm	$-2 \cdot 10^{10}$	Receiver clock drift relative to system time (relative frequency error)
TimeSystem	u1		255	Time system of which the offset is provided in this sub-block: 0: GPS time 1: Galileo time 3: GLONASS time
Datum	u1		255	This field defines in which datum the coordinates are expressed: 0: WGS84/ITRS 19: Datum equal to that used by the DGNSS/RTK base station 30: ETRS89 (ETRF2000 realization) 31: NAD83(2011), North American Datum (2011) 32: NAD83(PA11), North American Datum, Pacific plate (2011) 33: NAD83(MA11), North American Datum, Marianas plate (2011) 34: GDA94(2010), Geocentric Datum of Australia (2010) 250: First user-defined datum 251: Second user-defined datum
NrSV	u1		255	Total number of satellites used in the PVT computation.
WACorrInfo	u1		0	Bit field providing information about which wide area corrections have been applied: Bit 0: set if orbit and satellite clock correction information is used Bit 1: set if range correction information is used Bit 2: set if ionospheric information is used Bit 3: set if orbit accuracy information is used (UERE/SISA) Bit 4: set if DO229 Precision Approach mode is active Bits 5-7: Reserved
ReferenceID	u2		65535	This field indicates the reference ID of the differential information used. In case of DGPS or RTK operation, this field is to be interpreted as the base station identifier. In SBAS operation, this field is to be interpreted as the PRN of the geostationary satellite used (from 120 to 158). If multiple base stations or multiple geostationary satellites are used the value is set to 65534.
MeanCorrAge	u2	0.01 s	65535	In case of DGPS or RTK, this field is the mean age of the differential corrections. In case of SBAS operation, this field is the mean age of the 'fast corrections' provided by the SBAS satellites.

SignalInfo	u4		0	Bit field indicating the type of GNSS signals having been used in the PVT computations. If a bit i is set, the signal type having index i has been used. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.
AlertFlag	u1		0	<p>Bit field indicating integrity related information:</p> <p>Bits 0-1: RAIM integrity flag:</p> <ul style="list-style-type: none"> 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved <p>Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm)</p> <p>Bit 3: Reserved</p> <p>Bit 4: set if either the horizontal or the vertical 2DRMS accuracy is higher than the horizontal or vertical alert limits set by the <code>setNWALevels</code> command.</p> <p>Bits 5-7: Reserved</p>
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	<p>Bit field containing PPP-related information:</p> <p>Bits 0-11: Age of the last seed, in seconds. The age is clipped to 4091s. This field must be ignored when the seed type is 0 (see bits 13-15 below).</p> <p>Bit 12: Reserved</p> <p>Bits 13-15: Type of last seed:</p> <ul style="list-style-type: none"> 0: Not seeded or not in PPP positioning mode 1: Manual seed 2: Seeded from DGPS 3: Seeded from RTKFixed
Latency	u2	0.0001 s	65535	Reserved for future use
HAccuracy	u2	0.01 m	65535	2DRMS horizontal accuracy: twice the root-mean-square of the horizontal distance error. The horizontal distance between the true position and the computed position is expected to be lower than HAccuracy with a probability of at least 95%. The value is clipped to 65534 = 655.34m
VAccuracy	u2	0.01 m	65535	2DRMS vertical accuracy: twice the root-mean-square of the vertical error. The vertical distance between the true position and the computed position is expected to be lower than VAccuracy with a probability of at least 95%. The value is clipped to 65534 = 655.34m.

Rev 1

Rev 2

Rev 2

Misc	u1			<p>Bit field containing miscellaneous flags:</p> <p>Bit 0: In DGNSS or RTK mode, set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown.</p> <p>Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown.</p> <p>Bit 2: Proprietary.</p> <p>Bit 3: Proprietary.</p> <p>Bits 4-7: Reserved</p>
Padding	u1[..]			Padding bytes, see 3.2.5

PosCovCartesian	Number: 5905 "OnChange" interval: 10 ms
-----------------	--

This block contains the elements of the symmetric variance-covariance matrix of the position expressed relative to the Cartesian axes of the coordinate system datum requested by the user:

$$\begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} & \sigma_{xb} \\ \sigma_{yx} & \sigma_y^2 & \sigma_{yz} & \sigma_{yb} \\ \sigma_{zx} & \sigma_{zy} & \sigma_z^2 & \sigma_{zb} \\ \sigma_{bx} & \sigma_{by} & \sigma_{bz} & \sigma_b^2 \end{pmatrix}$$

This variance-covariance matrix contains an indication of the accuracy of the estimated parameters (see diagonal elements) and the correlation between these estimates (see off-diagonal elements). Note that the variances and covariances are estimated: they are not necessarily indicative of the actual scatter of the position estimates at a given site.

The position variance results from the propagation of all pseudorange variances using the observation geometry. The receiver implements a stochastic error model for individual measurements, based on parameters such as the C/N₀, the satellite elevation, the pseudorange type, the URA of the broadcast ephemeris and the ionospheric model.

If the ellipsoidal height is not estimated (2D-mode), all components of the variance-covariance matrix are undefined and set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)

Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) Bits 4-5: Reserved Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position. Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).
Error	u1			PVT error code. The following values are defined: 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.
Cov_xx	f4	1 m ²	-2 · 10 ¹⁰	Variance of the x estimate

Cov_yy	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the y estimate
Cov_zz	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the z estimate
Cov_bb	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the clock bias estimate
Cov_xy	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the x and y estimates
Cov_xz	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the x and z estimates
Cov_xb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the x and clock bias estimates
Cov_yz	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the y and z estimates
Cov_yb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the y and clock bias estimates
Cov_zb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the z and clock bias estimates
Padding	u1[..]			Padding bytes, see 3.2.5

PosCovGeodetic	Number: 5906
	"OnChange" interval: 10 ms

This block contains the elements of the symmetric variance-covariance matrix of the position expressed in the geodetic coordinates in the datum requested by the user:

$$\begin{pmatrix} \sigma_{\phi}^2 & \sigma_{\phi\lambda} & \sigma_{\phi h} & \sigma_{\phi b} \\ \sigma_{\lambda\phi} & \sigma_{\lambda}^2 & \sigma_{\lambda h} & \sigma_{\lambda b} \\ \sigma_{h\phi} & \sigma_{h\lambda} & \sigma_h^2 & \sigma_{hb} \\ \sigma_{b\phi} & \sigma_{b\lambda} & \sigma_{bh} & \sigma_b^2 \end{pmatrix}$$

Please refer to the `PosCovCartesian` block description for a general explanation of the contents.

Note that the units of measure for all the variances and covariances, for height as well as for latitude and longitude, are m² for ease of interpretation.

If the ellipsoidal height is not estimated (2D-mode), all height related components of the variance-covariance matrix are undefined and set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.

WNc	u2	1 week	65535	<p>The GPS week number associated with the <code>TOW</code>. <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks.</p> <p>By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.</p>
Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Cov_latlat	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the latitude estimate
Cov_lonlon	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the longitude estimate
Cov_hgthgt	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the height estimate
Cov_bb	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the clock-bias estimate
Cov_latlon	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the latitude and longitude estimates

Cov_lathgt	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the latitude and height estimates
Cov_latb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the latitude and clock-bias estimates
Cov_lonhgt	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the longitude and height estimates
Cov_lonb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the longitude and clock-bias estimates
Cov_hb	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the height and clock-bias estimates
Padding	u1[..]			Padding bytes, see 3.2.5

VelCovCartesian	Number: 5907 "OnChange" interval: 10 ms
-----------------	--

This block contains the elements of the symmetric variance-covariance matrix of the velocity expressed in the Cartesian coordinates of the coordinate system datum requested by the user:

$$\begin{pmatrix} \sigma_{v_x}^2 & \sigma_{v_x v_y} & \sigma_{v_x v_z} & \sigma_{v_x d} \\ \sigma_{v_y v_x} & \sigma_{v_y}^2 & \sigma_{v_y v_z} & \sigma_{v_y d} \\ \sigma_{v_z v_x} & \sigma_{v_z v_y} & \sigma_{v_z}^2 & \sigma_{v_z d} \\ \sigma_{dv_x} & \sigma_{dv_y} & \sigma_{dv_z} & \sigma_d^2 \end{pmatrix}$$

Please refer to the `PosCovCartesian` block description for a general explanation of the contents.

If the up-velocity is not estimated (2D-mode), all components of the variance-covariance matrix are undefined and set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Cov_VxVx	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the x-velocity estimate
Cov_VyVy	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the y-velocity estimate
Cov_VzVz	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the z-velocity estimate
Cov_DtDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the clock drift estimate
Cov_VxVy	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the x- and y-velocity estimates
Cov_VxVz	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the x- and z-velocity estimates
Cov_VxDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the x-velocity and the clock drift estimates
Cov_VyVz	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the y- and z-velocity estimates

Cov_VyDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the y-velocity and the clock drift estimates
Cov_VzDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the z-velocity and the clock drift estimates
Padding	u1[..]			Padding bytes, see 3.2.5

VelCovGeodetic	Number: 5908 "OnChange" interval: 10 ms
----------------	--

This block contains the elements of the symmetric variance-covariance matrix of the velocity expressed in the geodetic coordinates in the datum requested by the user:

$$\begin{pmatrix} \sigma_{v_N}^2 & \sigma_{v_N v_E} & \sigma_{v_N v_U} & \sigma_{v_N d} \\ \sigma_{v_E v_N} & \sigma_{v_E}^2 & \sigma_{v_E v_U} & \sigma_{v_E d} \\ \sigma_{v_U v_N} & \sigma_{v_U v_E} & \sigma_{v_U}^2 & \sigma_{v_U d} \\ \sigma_{dv_N} & \sigma_{dv_E} & \sigma_{dv_U} & \sigma_d^2 \end{pmatrix}$$

Please refer to the PosCovCartesian block description for a general explanation of the contents.

If the up-velocity is not estimated (2D-mode), all up-velocity related components of the variance-covariance matrix are undefined and set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Cov_VnVn	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the north-velocity estimate
Cov_VeVe	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the east-velocity estimate
Cov_VuVu	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the up-velocity estimate
Cov_DtDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Variance of the clock drift estimate
Cov_VnVe	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the north- and east-velocity estimates
Cov_VnVu	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the north- and up-velocity estimates
Cov_VnDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the north-velocity and clock drift estimates
Cov_VeVu	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the east- and up-velocity estimates

Cov_VeDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the east-velocity and clock drift estimates
Cov_VuDt	f4	$1 \text{ m}^2 / \text{s}^2$	$-2 \cdot 10^{10}$	Covariance between the up-velocity and clock drift estimates
Padding	u1[..]			Padding bytes, see 3.2.5

DOP	Number:	4001
	"OnChange" interval:	10 ms

This block contains both Dilution of Precision (DOP) values and SBAS protection levels. The DOP values result from a trace of the unit position variance-covariance matrices:

$$\text{Position Dilution of Precision: } PDOP = \sqrt{\mathbf{Q}_{xx} + \mathbf{Q}_{yy} + \mathbf{Q}_{zz}}$$

$$\text{Time Dilution of Precision: } TDOP = \sqrt{\mathbf{Q}_{bb}}$$

$$\text{Horizontal Dilution of Precision: } HDOP = \sqrt{\mathbf{Q}_{\lambda\lambda} + \mathbf{Q}_{\phi\phi}}$$

$$\text{Vertical Dilution of Precision: } VDOP = \sqrt{\mathbf{Q}_{hh}}$$

In these equations, the matrix \mathbf{Q} is the inverse of the unweighted normal matrix used for the computation of the position. The normal matrix equals the product of the geometry matrix A with its transpose ($A^t A$). The term "unweighted" implies that the DOP factor only addresses the effect of the geometric factors on the quality of the position.

The DOP values can be used to interpret the current constellation geometry. This is an important parameter for the quality of the position fix: the DOP parameter is the propagation factor of the pseudorange variance. For example, if an error of 5 m is present in the pseudorange, it will propagate into the horizontal plane with a factor expressed by the HDOP. Hence a low DOP value indicates that the satellites used for the position fix result in a low multiplication of the systematic ranging errors. A value of six (6) for the PDOP is generally considered as the maximum value allowed for an acceptable position computation.

The horizontal and vertical protection levels (HPL and VPL) indicate the integrity of the computed horizontal and vertical position components as per the DO 229 specification. In SBAS-aided PVT mode (see the `Mode` field of the `PVTCartesian` SBF block), HPL and VPL are based upon the error estimates provided by SBAS. Otherwise they are based upon internal position-mode dependent error estimates.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
NrSV	u1		0	Total number of satellites used in the DOP computation, or 0 if the DOP information is not available (in that case, the <code>x_{DOP}</code> fields are all set to 0)
Reserved	u1			Reserved for future use, to be ignored by decoding software
PDOP	u2	0.01	0	If 0, PDOP not available, otherwise divide by 100 to obtain PDOP.
TDOP	u2	0.01	0	If 0, TDOP not available, otherwise divide by 100 to obtain TDOP.
HDOP	u2	0.01	0	If 0, HDOP not available, otherwise divide by 100 to obtain HDOP.
VDOP	u2	0.01	0	If 0, VDOP not available, otherwise divide by 100 to obtain VDOP.
HPL	f4	1 m	$-2 \cdot 10^{10}$	Horizontal Protection Level (see the DO 229 standard).
VPL	f4	1 m	$-2 \cdot 10^{10}$	Vertical Protection Level (see the DO 229 standard).
Padding	u1[...]			Padding bytes, see 3.2.5

PosCart	Number: 4044 "OnChange" interval: 10 ms
---------	--

This block contains the absolute and relative (relative to the nearest base station) position at the time specified in the `TOW` and `WNc` fields. The time of applicability is specified in the receiver time frame.

The absolute position (X, Y, Z) is reported in a Cartesian coordinate system using the datum indicated in the `Datum` field. The position is that of the marker. The ARP-to-marker offset is set through the command `setAntennaOffset`.

For highest accuracy, the receiver tries to compute the baseline (`Base2RoverX`, `Base2RoverY`, `Base2RoverZ`) from rover ARP to base ARP. See the description of the `BaseVectorCart` block for details.

Accurate ARP-to-ARP baseline is guaranteed only if both bits 0 and 1 of the `Misc` field are set. Otherwise, centimeter-level offsets may arise because the receiver cannot make the distinction between phase center and ARP positions. See the Firmware User Manual for a discussion on the phase center and ARP positions.

This block also contains the variance-covariance information and DOP factors associated with the position.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.

WNc	u2	1 week	65535	<p>The GPS week number associated with the <code>TOW</code>. <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks.</p> <p>By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.</p>
Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
X	f8	1 m	$-2 \cdot 10^{10}$	Marker X coordinate in coordinate frame specified by <code>Datum</code>
Y	f8	1 m	$-2 \cdot 10^{10}$	Marker Y coordinate in coordinate frame specified by <code>Datum</code>
Z	f8	1 m	$-2 \cdot 10^{10}$	Marker Z coordinate in coordinate frame specified by <code>Datum</code>
Base2RoverX	f8	1 m	$-2 \cdot 10^{10}$	X baseline component (from base to rover)

Base2RoverY	f8	1 m	$-2 \cdot 10^{10}$	Y baseline component (from base to rover)
Base2RoverZ	f8	1 m	$-2 \cdot 10^{10}$	Z baseline component (from base to rover)
Cov_xx	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the x estimate
Cov_yy	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the y estimate
Cov_zz	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of the z estimate
Cov_xy	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the x and y estimates
Cov_xz	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the x and z estimates
Cov_yz	f4	1 m ²	$-2 \cdot 10^{10}$	Covariance between the y and z estimates
PDOP	u2	0.01	0	If 0, PDOP not available, otherwise divide by 100 to obtain PDOP.
HDOP	u2	0.01	0	If 0, HDOP not available, otherwise divide by 100 to obtain HDOP.
VDOP	u2	0.01	0	If 0, VDOP not available, otherwise divide by 100 to obtain VDOP.
Misc	u1			<p>Bit field containing miscellaneous flags:</p> <p>Bit 0: In DGNSS or RTK mode, set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown.</p> <p>Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown.</p> <p>Bit 2: Proprietary.</p> <p>Bit 3: Proprietary.</p> <p>Bits 4-7: Reserved</p>
Reserved	u1			Reserved for future use.
AlertFlag	u1		0	<p>Bit field indicating integrity related information:</p> <p>Bits 0-1: RAIM integrity flag:</p> <ul style="list-style-type: none"> 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved <p>Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm)</p> <p>Bit 3: Reserved</p> <p>Bit 4: set if either the horizontal or the vertical 2DRMS accuracy is higher than the horizontal or vertical alert limits set by the setNWAlevels command.</p> <p>Bits 5-7: Reserved</p>

Datum	u1		255	<p>This field defines in which datum the coordinates are expressed:</p> <ul style="list-style-type: none"> 0: WGS84/ITRS 19: Datum equal to that used by the DGNSS/RTK base station 30: ETRS89 (ETRF2000 realization) 31: NAD83(2011), North American Datum (2011) 32: NAD83(PA11), North American Datum, Pacific plate (2011) 33: NAD83(MA11), North American Datum, Marianas plate (2011) 34: GDA94(2010), Geocentric Datum of Australia (2010) 250: First user-defined datum 251: Second user-defined datum
NrSV	u1		255	Total number of satellites used in the PVT computation.
WACorrInfo	u1		0	<p>Bit field providing information about which wide area corrections have been applied:</p> <ul style="list-style-type: none"> Bit 0: set if orbit and satellite clock correction information is used Bit 1: set if range correction information is used Bit 2: set if ionospheric information is used Bit 3: set if orbit accuracy information is used (UERE/SISA) Bit 4: set if DO229 Precision Approach mode is active Bits 5-7: Reserved
ReferenceId	u2		65535	<p>This field indicates the reference ID of the differential information used.</p> <p>In case of DGPS or RTK operation, this field is to be interpreted as the base station identifier. In SBAS operation, this field is to be interpreted as the PRN of the geostationary satellite used (from 120 to 158). If multiple base stations or multiple geostationary satellites are used the value is set to 65534.</p>
MeanCorrAge	u2	0.01 s	65535	<p>In case of DGPS or RTK, this field is the mean age of the differential corrections.</p> <p>In case of SBAS operation, this field is the mean age of the 'fast corrections' provided by the SBAS satellites.</p>
SignalInfo	u4		0	<p>Bit field indicating the type of GNSS signals having been used in the PVT computations. If a bit i is set, the signal type having index i has been used. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of <code>SignalInfo</code>.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

PosLocal	Number: 4052 "OnChange" interval: 10 ms
----------	--

This block contains the position at the time specified in the `TOW` and `WNc` fields. The time of applicability is specified in the receiver time frame.

The position (Lat, Lon, Alt) relates to the local datum identified with the `Datum` field. The coordinate transformation to the local datum is done using parameters transmitted by the RTK service provider in RTCM message types MT1021 to MT1023.

The position is that of the marker. The ARP-to-marker offset is set through the command **setAntennaOffset**.

If no position is available, the `Error` field indicates the cause of the unavailability and all fields after the `Error` field are set to their respective Do-Not-Use values.

To be able to output a position in the `PosLocal` block, the receiver needs to have received the relevant RTCM transformation messages (at least either MT1021 or MT1022 is required). If they have not been received yet, the local position is not available and the `Error` field is set to value 17. See also section "Datum Transformation" in the Firmware User Manual.

The corresponding `RTCMDatum` block provides information on the local datum name and transformation quality indicators. The corresponding `RTCMDatum` block is the one of which the `Datum` field matches the `Datum` field in the `PosLocal` block.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.

TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	<p>The GPS week number associated with the TOW. WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks.</p> <p>By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.</p>
Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the Error field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command setPVTMode, base, auto and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions 17: Datum transformation parameters unknown <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Lat	f8	1 rad	$-2 \cdot 10^{10}$	Marker latitude, from $-\pi/2$ to $+\pi/2$, positive North of Equator

Lon	f8	1 rad	$-2 \cdot 10^{10}$	Marker latitude, from $-\pi$ to $+\pi$, positive North of Equator
Alt	f8	1 m	$-2 \cdot 10^{10}$	Marker height. See the <code>HeightType</code> field of the corresponding <code>RTCMDatum</code> block for the interpretation of the height.
Datum	u1			Reference frame to which the position relates. Internal ID of the local target datum from RTCMv3 MT1021/1022, from 20 to 24. The corresponding datum parameters can be found in the <code>RTCMDatum</code> block having a matching <code>Datum</code> field.
Padding	u1[.]			Padding bytes, see 3.2.5

PosProjected	Number: 4094
	"OnChange" interval: 10 ms

This block contains the projected coordinates at the time specified in the `TOW` and `WNc` fields. The time of applicability is specified in the receiver time frame.

The coordinates (Northing, Easting, Alt) relate to the local datum identified with the `Datum` field. The coordinate transformation and projection is done using parameters transmitted by the RTK service provider in RTCM message types MT1021 to MT1027.

The position is that of the marker. The ARP-to-marker offset is set through the command `setAntennaOffset`.

If no position is available, the `Error` field indicates the cause of the unavailability and all fields after the `Error` field are set to their respective Do-Not-Use values.

To be able to output a position in the `PosProjected` block, the receiver needs to have received at least one RTCM message in the MT1025 to MT1027 range. If none of these messages is sent out by the service provider, or if they have not been received yet, the projected position is not available and the `Error` field is set to value 17. See also section "Datum Transformation" in the Firmware User Manual.

The corresponding `RTCMDatum` block provides information on the local datum name and transformation/projection quality indicators. The corresponding `RTCMDatum` block is the one of which the `Datum` field matches the `Datum` field in the `PosProjected` block.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions 17: Datum transformation parameters unknown <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Northing	f8	1 m	$-2 \cdot 10^{10}$	Marker northing coordinate in the plane grid representation.
Easting	f8	1 m	$-2 \cdot 10^{10}$	Marker easting coordinate in the plane grid representation.
Alt	f8	1 m	$-2 \cdot 10^{10}$	Marker height. If the <code>Datum</code> field is in the 20 to 24 range (it is always the case in the current firmware), see the <code>HeightType</code> field of the corresponding <code>RTCDatum</code> block for the interpretation of the height.

Datum	u1			Reference frame to which the coordinates relate. If the value is in the 20 to 24 range (it is always the case in the current firmware), the corresponding datum parameters can be found in the <code>RTCMDatum</code> block having a matching <code>Datum</code> field.
Padding	u1[..]			Padding bytes, see 3.2.5

PVTSatCartesian	Number: 4008 "OnChange" interval: 10 ms
-----------------	--

This block contains the position and velocity of all the satellites used in the PVT solution, together with slant ionospheric and tropospheric delays. Coordinates are referred to the time of signal transmission computed by the receiver and are corrected for the Sagnac effect.

The reference frame the coordinates are related to is the one specified in the respective ICDs (WGS84 for GPS satellites, GTRF for Galileo satellites, PZ90 for GLONASS satellites, etc).

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of satellites for which satellite position is provided in this SBF block, i.e. number of <code>SatPos</code> sub-blocks. If <code>N</code> is 0, there are no satellite positions available for this epoch.
SBlockLength	u1	1 byte		Length of one sub-block
SatPos		A succession of <code>N</code> <code>SatPos</code> sub-blocks, see definition below

Padding	u1[.]			Padding bytes, see 3.2.5
---------	-------	--	--	--------------------------

SatPos sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			Satellite ID, see 3.2.9
FreqNr	u1		0	For GLONASS satellites, this is the frequency number, with an offset of 8. It ranges from 1 (corresponding to an actual frequency number of -7) to 21 (corresponding to an actual frequency number of 13). For non-GLONASS satellites, FreqNr is reserved and must be ignored by the decoding software.
IODE	u2			IODE of the data set used to compute the values in this sub-block.
x	f8	1 m	$-2 \cdot 10^{10}$	X coordinate
y	f8	1 m	$-2 \cdot 10^{10}$	Y coordinate
z	f8	1 m	$-2 \cdot 10^{10}$	Z coordinate
Vx	f4	1 m / s	$-2 \cdot 10^{10}$	Satellite velocity in the X direction
Vy	f4	1 m / s	$-2 \cdot 10^{10}$	Satellite velocity in the Y direction
Vz	f4	1 m / s	$-2 \cdot 10^{10}$	Satellite velocity in the Z direction
IonoMSB	i2	1 dm	$-32768^{(7)}$	Total slant ionospheric delay at the L1 carrier frequency (1575.42MHz), with a decimeter resolution.
TropoMSB	i2	1 dm	$-32768^{(8)}$	Total slant tropospheric delay, with a decimeter resolution.
IonoLSB	u1	1.0/256.0 dm	0 ⁽⁷⁾	Sub-decimeter part of the slant ionospheric delay. The high-resolution ionospheric delay, expressed in meters, can be computed as: $\text{IonoDelay[m]} = 0.1 * (\text{IonoMSB} + \text{IonoLSB}/256)$
TropoLSB	u1	1.0/256.0 dm	0 ⁽⁸⁾	Sub-decimeter part of the slant tropospheric delay. The high-resolution tropospheric delay, expressed in meters, can be computed as: $\text{TropoDelay[m]} = 0.1 * (\text{TropoMSB} + \text{TropoLSB}/256)$
IonoModel	u1			Model used to compute the ionospheric delay: 0: Not applicable 1: Klobuchar 2: DO229 3: NeQuick 4: Measured (from dual frequency measurements) 5: Estimated
Padding	u1[.]			Padding bytes, see 3.2.5

Rev 1

⁽⁷⁾ The ionospheric delay should not be used when IonoMSB is -32768.

⁽⁸⁾ The tropospheric delay should not be used when TropoMSB is -32768.

PVTResiduals	Number: 4009 "OnChange" interval: 10 ms
--------------	--

This block contains the residuals of all measurements used in PVT solution computed at the epoch specified in the `TOW` and `WNc` fields. The PVT solution itself can be found in the `PVTCartesian` or `PVTGeodetic` blocks.

For each measurement from each satellite and each modulation used in the PVT solution, detailed PVT residual information is output for each observable type (code phase, carrier phase and Doppler):

- a-posteriori measurement residual (e_i)
- absolute value of the w -test statistic (w_i)
- Minimal detectable bias (MDB).

In case of multi-base differential operation, a set of residuals is provided for all base stations.

This block uses a two-level sub-block structure analogous to that of the `MeasEpoch` block. It contains one `SatSignalInfo` sub-block for each satellite/signal type pair used in the PVT or attitude computation. Each `SatSignalInfo` sub-block contains a number of `ResidualInfo` sub-blocks, each of them containing the residuals of a given observable type.

The standard deviation of the residual (σ_e) for satellite i and the "a priori" measurement standard deviation (σ_y) can be computed from e_i , w_i and MDB by using the following formulas (see also the Firmware User Manual):

$$\sigma_{e_i} = \frac{|e_i|}{w_i} \text{ and } \sigma_{y_i} = \sqrt{\frac{MDB}{\sqrt{\lambda_0}}} \cdot \sigma_{e_i}$$

where λ_0 is the non-centrality parameter and:

$$\sqrt{\lambda_0} = \sqrt{2}[\text{erfinv}(1 - P_{fa}) + \text{erfinv}(1 - 2P_{md})]$$

with P_{fa} and P_{md} being the probability of false alarm and of missed detection respectively, as set by the `setRAIMLevels` command, and the "erfinv" function being the inverse error function. The output of `erfinv(x)` is the value y that satisfies the following equality:

$$x = \frac{2}{\sqrt{\pi}} \int_0^y e^{-t^2} dt$$

This block can be used to monitor the quality of the measurements. Under normal circumstances, the residuals lie within -2 and +2 times the a-priori variance of the corresponding measurements.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of satellite/signal pairs for which residual blocks are provided in this SBF block, i.e. number of <code>SatSignalInfo</code> sub-blocks. If <code>N</code> is 0, there are no satellite residuals available for this epoch.
SB1Length	u1	1 byte		Length of a <code>SatSignalInfo</code> sub-block, excluding the nested <code>ResidualInfoCode</code> , <code>ResidualInfoPhase</code> and <code>ResidualInfoDoppler</code> sub-blocks
SB2Length	u1	1 byte		Length of a <code>ResidualInfoCode</code> , <code>ResidualInfoPhase</code> and <code>ResidualInfoDoppler</code> sub-block
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
Residuals		A succession of <code>N</code> <code>SatSignalInfo</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

SatSignalInfo sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			Satellite ID, see 3.2.9
FreqNr	u1		0	For GLONASS satellites, this is the frequency number, with an offset of 8. It ranges from 1 (corresponding to an actual frequency number of -7) to 21 (corresponding to an actual frequency number of 13). For non-GLONASS satellites, FreqNr is reserved and must be ignored by the decoding software.
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number as defined in 3.2.10. Bits 5-7: Antenna ID: 0 for the main antenna
RefSVID	u1		255, 62	Satellite ID of the reference satellite used for double differencing, see 3.2.9. Set to 255 if not in double difference mode, or set to 62 if in double difference mode, and GLONASS slot number unknown.
RefFreqNr	u1		255, 0	GLONASS frequency number for the reference satellite, see 3.2.9. Set to 255 if not in double difference mode, or set to 0 if in double difference mode, but non-GLONASS satellite.
MeasInfo	u1			Bit field: Bits 0-1: Type of residual this sub-block refers to: 0: zero-difference residual (standalone) 1: single-difference residual (SBAS, DGPS) 2: double-difference residual. If the antenna ID is 0 (see the Type field above), this sub-block contains an RTK residual, else it contains an attitude residual. Bit 2: Set if a ResidualInfoCode sub-block containing pseudorange residuals follows. Bit 3: Set if a ResidualInfoPhase sub-block containing carrier-phase residuals follows. Bit 4: Set if a ResidualInfoDoppler sub-block containing Doppler residuals follows. Bits 5-6: Reserved Bit 7: Set if ambiguity is fixed for the signal type identified by the Type field. The number of ResidualInfo sub-blocks to follow is equal to the number of bits set to 1 between bit 2 and bit 4. The order of these ResidualInfo sub-blocks is fixed: the code-phase residuals come first (if any), then the carrier phase residuals (if any), and the Doppler residuals as last.
IODE	u2			Issue of Data Ephemeris used for the satellite and signal type identified by SVID and Type.

Rev 1

CorrAge	u2	0.001 s	65535	Age of corrections, either from SBAS, DGPS, RTK etc, truncated to 655.34 seconds.
ReferenceID	u2		65535	ID of the base station the residuals apply to. Set to 65535 in case of standalone operation.
Padding	u1[..]			Padding bytes, see 3.2.5
If the Pseudorange residuals field is 1 then this sub block is available:				
ResidualInfoCode			A ResidualInfoCode sub-block, see definition below
If the Carrier-phase residuals field is 1 then this sub block is available:				
ResidualInfoPhase...	...			A ResidualInfoPhase sub-block, see definition below
If the Doppler residuals field is 1 then this sub block is available:				
ResidualInfoDoppler	...			A ResidualInfoDoppler sub-block, see definition below

ResidualInfoCode sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Residual	f4	1 m	$-2 \cdot 10^{10}$	Code Residual with respect to PVT solution reported in PVTCartesian and/or PVTGeodetic block.
W	u2	0.001	65535	Absolute value of the w -test statistic based on probability of false alarm set by user
MDB	u2	0.1 m	65535	Minimal detectable bias based on probability of missed detection set by user
Padding	u1[..]			Padding bytes, see 3.2.5

ResidualInfoPhase sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Residual	f4	1 cycle	$-2 \cdot 10^{10}$	Phase Residual with respect to PVT solution reported in PVTCartesian and/or PVTGeodetic block. Double-difference carrier phase residuals include the double difference ambiguity as long as the ambiguity is not fixed (i.e. as long as bit 7 of MeasInfo is not set). When the ambiguity is fixed, e_i does not contain the ambiguity anymore.
W	u2	0.001	65535	Absolute value of the w -test statistic based on probability of false alarm set by user
MDB	u2	0.01 cycles	65535	Minimal detectable bias based on probability of missed detection set by user
Padding	u1[..]			Padding bytes, see 3.2.5

ResidualInfoDoppler sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Residual	f4	1 m / s	$-2 \cdot 10^{10}$	Doppler Residual with respect to PVT solution reported in PVTCartesian and/or PVTGeodetic block.
W	u2	0.001	65535	Absolute value of the w -test statistic based on probability of false alarm set by user
MDB	u2	0.01 m / s	65535	Minimal detectable bias based on probability of missed detection set by user
Padding	u1[..]			Padding bytes, see 3.2.5

RAIMStatistics	Number: 4011 "OnChange" interval: 10 ms
----------------	--

This block contains the integrity statistics that are computed by the receiver RAIM algorithm.

The output of the RAIM algorithm contains integrity information, which can be used in user applications. First, the RAIM algorithm generates its own integrity flag based on the probability of false-alarm, which can be used by a user as a receiver-level indication of positional integrity. If the internal integrity test is successful, a user has an opportunity to introduce a more stringent application-specific integrity criterion by using External Reliability Levels (XERL). The positional solution is deemed as passed an application-level integrity test if the XERLs are within user-defined (and application-dependent) alarm limits. This comparison (and the definition of alarm limits as well) takes place in a user application and is outside of the receiver scope. Please refer to the RAIM section of the Firmware User Manual for further details.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

IntegrityFlag	u1			RAIM integrity flag: 0: Integrity test successful 1: Integrity test failed 2: Integrity not available
Reserved1	u1			Reserved for future use, to be ignored by decoding software
HERL-position	f4	1 m	$-2 \cdot 10^{10}$	Horizontal external reliability level of the position
VERL-position	f4	1 m	$-2 \cdot 10^{10}$	Vertical external reliability level of the position
HERL-velocity	f4	1 m / s	$-2 \cdot 10^{10}$	Horizontal external reliability level of the velocity
VERL-velocity	f4	1 m / s	$-2 \cdot 10^{10}$	Vertical external reliability level of the velocity
OverallModel	u2	1/50000	65535 ⁽⁹⁾	Overall model test statistic for the estimated PVT parameters divided by the test threshold
Padding	u1[..]			Padding bytes, see 3.2.5

⁽⁹⁾ This field is clipped to 65534, i.e. if the actual value is larger than 65534, it is set to 65534.

GEOCorrections	Number: 5935 "OnChange" interval: 10 ms
----------------	--

This block contains the SBAS corrections that the receiver has applied to the pseudoranges used in the PVT computation computed at the epoch specified in the `TOW` and `WNc` fields. The PVT solution itself can be found in the `PVTCartesian` or `PVTGeodetic` blocks.

The corrections are based on the messages received from an SBAS satellite. They compensate for the following errors:

- Satellite orbit
- Satellite clock
- Ionospheric delay.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of satellites for which corrections are provided in this SBF block, i.e. number of <code>SatCorr</code> sub-blocks. If <code>N</code> is 0, there are no corrections available for this epoch.

SBLength	u1	1 byte		Length of one sub-block in bytes
SatCorr		A succession of N SatCorr sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

SatCorr sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			Satellite ID, see 3.2.9
IODE	u1			Issue of Data Ephemeris related to the orbit and clock corrections
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software
PRC	f4	1 m		Applied pseudorange correction based on the fast correction data received in MT02-MT05 or MT24
CorrAgeFC	f4	1 s		Age of applied fast correction
DeltaX	f4	1 m		X-component of applied orbit correction based on the long term correction data received in MT24 or MT25
DeltaY	f4	1 m		Y-component of applied orbit correction based on the long term correction data received in MT24 or MT25
DeltaZ	f4	1 m		Z-component of applied orbit correction based on the long term correction data received in MT24 or MT25
DeltaClock	f4	1 s		Satellite clock correction based on the long term correction data received in MT24 or MT25
CorrAgeLT	f4	1 s		Age of applied long term correction
IonoPPlat	f4	1 rad	$-2 \cdot 10^{10}$	Latitude of ionospheric pierce point
IonoPPlon	f4	1 rad	$-2 \cdot 10^{10}$	Longitude of ionospheric pierce point
SlantIono	f4	1 m	$-2 \cdot 10^{10}$	Slant ionospheric delay at the L1 carrier at the ionosphere pierce point based on the data received in MT18 and MT26
CorrAgeIono	f4	1 s	$-2 \cdot 10^{10}$	Maximum of the ionospheric correction age at each of the grid locations used for the interpolation of the ionospheric delay.
VarFLT	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of fast and long-term corrections (used for XPL computation)
VarUIRE	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of ionospheric delay corrections (used for XPL computation)
VarAir	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of unmodeled receiver errors, such as tracking noise and multipath (used for XPL computation)
VarTropo	f4	1 m ²	$-2 \cdot 10^{10}$	Variance of tropospheric delay corrections (used for XPL computation)
Padding	u1[...]			Padding bytes, see 3.2.5

BaseVectorCart	Number: 4043
	"OnChange" interval: 10 ms

The `BaseVectorCart` block contains the relative position and orientation of one or more base stations, as seen from the rover (i.e. this receiver). The relative position is expressed in the Cartesian X, Y, Z directions.

For highest accuracy, the receiver tries to compute the baseline from rover antenna reference point (ARP) to base ARP. This requires to compensate for the phase center variation at both the base and the rover antennas. This is possible if two conditions are met:

- the base station must transmit its antenna parameters in RTCM2 message types 23 and 24 or in RTCM3 message types 1005/1006 and 1007/1008. Older RTCM2 messages and CMR do not allow phase center variation compensation.
- the base and rover antenna types must belong to the list returned by the command `1stAntennaInfo, overview`. (see the description of the commands `setAntennaOffset` and `1stAntennaInfo` for details).

Accurate ARP-to-ARP baseline is guaranteed only if both bits 0 and 1 of the `Misc` field are set. Otherwise, centimeter-level offsets may arise because the receiver cannot make the distinction between phase center and ARP positions. See the Firmware User Manual for a discussion on the phase center and ARP positions.

The block supports multi-base operation. It contains as many sub-blocks as available base stations, each sub-block containing the baseline relative to a single base station identified by the `ReferenceID` field.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <i>Sync1</i> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <i>Sync2</i> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <i>CRC</i> field is the 16-bit CRC of all the bytes in an SBF block from and including the <i>ID</i> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <i>ID</i> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <i>Length</i> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <i>TOW</i> . <i>WNc</i> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <i>WNc</i> is also the Galileo week number plus 1024.
N	u1			Number of baselines for which relative position, velocity and direction are provided in this SBF block, i.e. number of <i>VectorInfoCart</i> sub-blocks. If <i>N</i> is 0, there are no baseline available for this epoch.
SBLength	u1	1 byte		Length of one sub-block
<i>VectorInfoCart</i>		A succession of <i>N</i> <i>VectorInfoCart</i> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

VectorInfoCart sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
nrSV	u1			Number of satellites for which corrections are available from the base station identified by the ReferenceID field.
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the Error field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode,base,auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>

Misc	u1			<p>Bit field containing miscellaneous flags:</p> <p>Bit 0: Set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown.</p> <p>Bit 1: Set if the phase center variation is compensated for at the rover (i.e. the baseline starts from the antenna ARP), unset if not or unknown.</p> <p>Bit 2: Proprietary.</p> <p>Bits 3-7: Reserved</p>
DeltaX	f8	1 m	$-2 \cdot 10^{10}$	X baseline component (from rover to base)
DeltaY	f8	1 m	$-2 \cdot 10^{10}$	Y baseline component (from rover to base)
DeltaZ	f8	1 m	$-2 \cdot 10^{10}$	Z baseline component (from rover to base)
DeltaVx	f4	1 m / s	$-2 \cdot 10^{10}$	X velocity of base with respect to rover
DeltaVy	f4	1 m / s	$-2 \cdot 10^{10}$	Y velocity of base with respect to rover
DeltaVz	f4	1 m / s	$-2 \cdot 10^{10}$	Z velocity of base with respect to rover
Azimuth	u2	0.01 degrees	65535	Azimuth of the base station (from 0 to 360°, increasing towards east)
Elevation	i2	0.01 degrees	-32768	Elevation of the base station (from -90° to 90°)
ReferenceID	u2			Base station ID
CorrAge	u2	0.01 s	65535	Age of the oldest differential correction used for this baseline computation.
SignalInfo	u4		0	Bit field indicating the GNSS signals for which differential corrections are available from the base station identified by <code>ReferenceID</code> . If bit i is set, corrections for the signal type having index i are available. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of <code>SignalInfo</code> .
Padding	u1[...]			Padding bytes, see 3.2.5

BaseVectorGeod	Number: 4028
	"OnChange" interval: 10 ms

The `BaseVectorGeod` block contains the relative position and orientation of one or more base stations, as seen from the rover (i.e. this receiver). The relative position is expressed in the East-North-Up directions.

For highest accuracy, the receiver tries to compute the baseline from rover antenna reference point (ARP) to base ARP. See the description of the `BaseVectorCart` block for details.

Accurate ARP-to-ARP baseline is guaranteed only if both bits 0 and 1 of the `Misc` field are set. Otherwise, centimeter-level offsets may arise because the receiver cannot make the distinction between phase center and ARP positions. See the Firmware User Manual for a discussion on the phase center and ARP positions.

The block supports multi-base operation. It contains as many sub-blocks as available base stations, each sub-block containing the baseline coordinates relative to a single base station identified by the `ReferenceID` field.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <i>Sync1</i> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <i>Sync2</i> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <i>CRC</i> field is the 16-bit CRC of all the bytes in an SBF block from and including the <i>ID</i> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <i>ID</i> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <i>Length</i> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <i>TOW</i> . <i>WNc</i> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <i>WNc</i> is also the Galileo week number plus 1024.
N	u1			Number of baselines for which relative position, velocity and direction are provided in this SBF block, i.e. number of <i>VectorInfoGeod</i> sub-blocks. If <i>N</i> is 0, there are no baseline available for this epoch.
SBLength	u1	1 byte		Length of one sub-block
<i>VectorInfoGeod</i>		A succession of <i>N</i> <i>VectorInfoGeod</i> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

VectorInfoGeod sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
NrSV	u1			Number of satellites for which corrections are available from the base station identified by the <code>ReferenceID</code> field.
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode,base,auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>

Misc	u1			<p>Bit field containing miscellaneous flags:</p> <p>Bit 0: Set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown.</p> <p>Bit 1: Set if the phase center variation is compensated for at the rover (i.e. the baseline starts from the antenna ARP), unset if not or unknown.</p> <p>Bit 2: Proprietary.</p> <p>Bits 3-7: Reserved</p>
DeltaEast	f8	1 m	$-2 \cdot 10^{10}$	East baseline component (from rover to base)
DeltaNorth	f8	1 m	$-2 \cdot 10^{10}$	North baseline component (from rover to base)
DeltaUp	f8	1 m	$-2 \cdot 10^{10}$	Up baseline component (from rover to base)
DeltaVe	f4	1 m / s	$-2 \cdot 10^{10}$	East velocity of base with respect to rover
DeltaVn	f4	1 m / s	$-2 \cdot 10^{10}$	North velocity of base with respect to rover
DeltaVu	f4	1 m / s	$-2 \cdot 10^{10}$	Up velocity of base with respect to rover
Azimuth	u2	0.01 degrees	65535	Azimuth of the base station (from 0 to 360°, increasing towards east)
Elevation	i2	0.01 degrees	-32768	Elevation of the base station (from -90° to 90°)
ReferenceID	u2			Base station ID
CorrAge	u2	0.01 s	65535	Age of the oldest differential correction used for this baseline computation.
SignalInfo	u4		0	Bit field indicating the GNSS signals for which differential corrections are available from the base station identified by ReferenceID. If bit i is set, corrections for the signal type having index i are available. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.
Padding	u1[.]			Padding bytes, see 3.2.5

PVTSupport	Number: 4076
	"OnChange" interval: 10 ms

This block is undocumented. It is for maintenance purpose only.

EndOfPVT	Number: 5921 "OnChange" interval: 10 ms
----------	--

This block marks the end of transmission of all PVT related blocks belonging to the same epoch.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Padding	u1[...]			Padding bytes, see 3.2.5

3.3.10 GNSS Attitude Blocks

AttEuler	Number: 5938 "OnChange" interval: 10 ms
----------	--

The `AttEuler` block contains the Euler angles (pitch, roll and heading) at the time specified in the `TOW` and `WNc` fields (in the receiver time frame).

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
NrSV	u1		255	The average over all antennas of the number of satellites currently included in the attitude calculations.
Error	u1			Bit field providing error information. For each antenna baseline, two bits are used to provide error information: Bits 0-1: Error code for Main-Aux1 baseline Bits 2-3: Error code for Main-Aux2 baseline Bits 4-6: Reserved Bit 7: Set when attitude not requested by user (see command <code>setGNSSAttitude</code>). In that case, the other bits are all zero. The error codes per antenna are: 00b: no error 01b: not enough measurements 10b: antennas are aligned 11b: Inconsistency with manual antenna position information

Mode	u2			Attitude mode code: 0: No attitude 1: Heading, pitch (roll = 0), aux antenna positions obtained with float ambiguities 2: Heading, pitch (roll = 0), aux antenna positions obtained with fixed ambiguities 3: Heading, pitch, roll, aux antenna positions obtained with float ambiguities 4: Heading, pitch, roll, aux antenna positions obtained with fixed ambiguities
Reserved	u2			Reserved for future use, to be ignored by decoding software
Heading	f4	1 degree	$-2 \cdot 10^{10}$	Heading
Pitch	f4	1 degree	$-2 \cdot 10^{10}$	Pitch
Roll	f4	1 degree	$-2 \cdot 10^{10}$	Roll
PitchDot	f4	1 degree / s	$-2 \cdot 10^{10}$	Rate of change of the pitch angle
RollDot	f4	1 degree / s	$-2 \cdot 10^{10}$	Rate of change of the roll angle
HeadingDot	f4	1 degree / s	$-2 \cdot 10^{10}$	Rate of change of the heading angle
Padding	u1[..]			Padding bytes, see 3.2.5

AttCovEuler	Number: 5939 "OnChange" interval: 10 ms
-------------	--

This block contains the elements of the symmetric variance-covariance matrix of the attitude angles reported in the `AttEuler` block

$$\begin{pmatrix} \sigma_{\phi}^2 & \sigma_{\phi\theta} & \sigma_{\phi\psi} \\ \sigma_{\theta\phi} & \sigma_{\theta}^2 & \sigma_{\theta\psi} \\ \sigma_{\psi\phi} & \sigma_{\psi\theta} & \sigma_{\psi}^2 \end{pmatrix}$$

This variance-covariance matrix contains an indication of the accuracy of the estimated parameters (see diagonal elements) and the correlation between these estimates (see off-diagonal elements).

In case the receiver is in heading and pitch mode only, only the heading and pitch variance values will be valid. All other components of the variance-covariance matrix are set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Reserved	u1			Reserved for future use, to be ignored by decoding software
Error	u1			Bit field providing error information. For each antenna baseline, two bits are used to provide error information: Bits 0-1: Error code for Main-Aux1 baseline Bits 2-3: Error code for Main-Aux2 baseline Bits 4-6: Reserved Bit 7: Set when attitude not requested by user (see command <code>setGNSSAttitude</code>). In that case, the other bits are all zero. The error codes per antenna are: 00b: no error 01b: not enough measurements 10b: antennas are aligned 11b: Inconsistency with manual antenna position information
Cov_HeadHead	f4	1 degree ²	-2 · 10 ¹⁰	Variance of the heading estimate

Cov_PitchPitch	f4	1 degree ²	$-2 \cdot 10^{10}$	Variance of the pitch estimate
Cov_RollRoll	f4	1 degree ²	$-2 \cdot 10^{10}$	Variance of the roll estimate
Cov_HeadPitch	f4	1 degree ²	$-2 \cdot 10^{10}$	Covariance between Euler angle estimates. Future functionality. The values are currently set to their Do-Not-Use values.
Cov_HeadRoll	f4	1 degree ²	$-2 \cdot 10^{10}$	Covariance between Euler angle estimates. Future functionality. The values are currently set to their Do-Not-Use values.
Cov_PitchRoll	f4	1 degree ²	$-2 \cdot 10^{10}$	Covariance between Euler angle estimates. Future functionality. The values are currently set to their Do-Not-Use values.
Padding	u1[..]			Padding bytes, see 3.2.5

EndOfAtt	Number: 5943 "OnChange" interval: 10 ms
----------	--

This block marks the end of transmission of all GNSS-attitude related blocks belonging to the same epoch.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Padding	u1[...]			Padding bytes, see 3.2.5

3.3.11 Receiver Time Blocks

ReceiverTime	Number: 5914 "OnChange" interval: 1s
--------------	---

The `ReceiverTime` block provides the current time with a 1-second resolution in the receiver time scale and UTC.

The level of synchronization of the receiver time with the satellite system time is provided in the `SyncLevel` field.

UTC time is provided if the UTC parameters have been received from at least one GNSS satellite. If the UTC time is not available, the corresponding fields are set to their Do-Not-Use value.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
UTCYear	i1	1 year	-128	Current year in the UTC time scale (2 digits). From 0 to 99, or -128 if not available

UTCMonth	i1	1 month	–128	Current month in the UTC time scale. From 1 to 12, or -128 if not available
UTCDay	i1	1 day	–128	Current day in the UTC time scale. From 1 to 31, or -128 if not available
UTCHour	i1	1 hour	–128	Current hour in the UTC time scale. From 0 to 23, or -128 if not available
UTCMin	i1	1 minute	–128	Current minute in the UTC time scale. From 0 to 59, or -128 if not available
UTCSec	i1	1 s	–128	Current second in the UTC time scale. From 0 to 59, or -128 if not available
DeltaLS	i1	1 s	–128	Integer second difference between UTC time and GPS system time. Positive if GPS time is ahead of UTC. Set to -128 if not available.
SyncLevel	u1			<p>Bit field indicating the synchronization level of the receiver time. If bits 0 to 2 are set, full synchronization is achieved:</p> <p>Bit 0: WNSET: if this bit is set, the receiver week number is set.</p> <p>Bit 1: TOWSET: if this bit is set, the receiver time-of-week is set to within 20ms.</p> <p>Bit 2: FINETIME: if this bit is set, the receiver time-of-week is within the limit specified by the setClockSyncThreshold command.</p> <p>Bits 3-7: Reserved</p>
Padding	u1[.]			Padding bytes, see 3.2.5

xPPSOffset	Number: 5911 "OnChange" interval: PPS rate
------------	---

The `xPPSOffset` block contains the offset between the true xPPS pulse and the actual pulse output by the receiver. It is output right after each xPPS pulse.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SyncAge	u1	1 s		Age of the last synchronization to system time. The xPPS pulse is regularly resynchronized with system time. This field indicates the number of seconds elapsed since the last resynchronization. <code>SyncAge</code> is constrained to the 0-255s range. If the age is higher than 255s, <code>SyncAge</code> is set to 255. If the PPS is synchronized with the internal receiver time (<code>Timescale</code> = 3), <code>SyncAge</code> is always set to 0.

TimeScale	u1			Time reference to which the xPPS pulse is referenced. The following values are defined (see also the setPPSPParameters command): 1: GNSS system time specified by the setTimingSystem command 2: UTC 3: receiver time 4: GLONASS time
Offset	f4	$1 \cdot 10^{-9} \text{ s}$		Offset of the xPPS output by the receiver with respect to its true position. <i>Offset</i> is negative when the xPPS pulse is in advance with respect to its true position. See the Firmware User Manual for an explanation of the xPPS generation principle, and for a description of the xPPS offset.
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.12 External Event Blocks

These blocks report the state of the receiver applicable at the instant of a level transition on one of its “Event” pins. The receiver time is reported in the `ExtEvent` SBF block, and the receiver position is reported in the `ExtEventPVTCartesian` and the `ExtEventPVTGeodetic` blocks.

If enabled, upon detection of an event, these three blocks are output in the following order, with no other SBF blocks in between them:

1. `ExtEvent`;
2. `ExtEventPVTCartesian`;
3. `ExtEventPVTGeodetic`.

All blocks referring to the same event contain the same time stamp in the `TOW` and `WNc` fields.

ExtEvent	Number: 5924
	"OnChange" interval: each time an event is detected

The ExtEvent block contains the time tag of a voltage transition on one of the "Event" input pins.

This block is only output after the first position fix is available.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Source	u1			Input pin where this external event has been detected. The following values are defined: 1: EventA 2: EventB
Polarity	u1			0: rising edge event 1: falling edge event
Offset	f4	1 s		Event time offset with respect to <code>TOW</code> , including the potential delay specified with the <code>setEventParameters</code> command. The time of week of the external event is given by: $t_{\text{ext},x} [\text{s}] = \text{TOW}/1000 + \text{Offset}$ <code>t_{ext,x}</code> refers to the receiver system time scale. Use the <code>RxClkBias</code> field to convert this time to the GNSS time scale.

RxClkBias	f8	1 s	$-2 \cdot 10^{10}$	<p>Receiver clock bias at the time of event. The clock bias is relative to the system time specified by the setTimingSystem command. To get the time of week of the external event in that system time scale, use:</p> $t_{\text{ext,GNSS}} [\text{s}] = \text{TOW}/1000 + \text{Offset} - \text{RxClkBias}.$ <p>The accuracy of the clock bias is dependent on the age of the last PVT solution. When the receiver has been unable to compute a PVT during the last 10 minutes, this field is set to its Do-Not-Use value.</p>
PVTAge	u2	1 s		Age of the last PVT solution. If the PVT age is larger than 10 minutes (600s), this value is clipped to 600.
Padding	u1[..]			Padding bytes, see 3.2.5

Rev 1

ExtEventPVTCartesian	Number:	4037
	"OnChange" interval:	each time an external event is detected

This block contains the position, velocity and time (PVT) solution applicable at the time of an external event, in a Cartesian coordinate system.

This block has the same structure and description as the `PVTCartesian` block, except that the `TOW` and `WNc` fields refer to the time at which the electrical transition on the event pin has been detected (with a millisecond resolution), and that the position is computed at the event time, taking into account a possible user-defined delay set by the `setEventParameters` command.

A user needing the sub-millisecond part of the event time must refer to the `Offset` field of the corresponding `ExtEvent` block. The corresponding `ExtEvent` block is the last of the `ExtEvent` blocks having been output by the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
X	f8	1 m	$-2 \cdot 10^{10}$	Marker X coordinate in coordinate frame specified by <code>Datum</code>
Y	f8	1 m	$-2 \cdot 10^{10}$	Marker Y coordinate in coordinate frame specified by <code>Datum</code>
Z	f8	1 m	$-2 \cdot 10^{10}$	Marker Z coordinate in coordinate frame specified by <code>Datum</code>
Undulation	f4	1 m	$-2 \cdot 10^{10}$	Geoid undulation computed from the global geoid model defined in the document 'Technical Characteristics of the NAVSTAR GPS, NATO, June 1991'
Vx	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the X direction
Vy	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the Y direction
Vz	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the Z direction

COG	f4	1 degree	$-2 \cdot 10^{10}$	Course over ground: this is defined as the angle of the vehicle with respect to the local level North, ranging from 0 to 360, and increasing towards east. Set to the do-not-use value when the speed is lower than 0.1m/s.
RxClkBias	f8	1 ms	$-2 \cdot 10^{10}$	Receiver clock bias relative to system time reported in the <code>TimeSystem</code> field. To transfer the receiver time to the system time, use: $t_{GPS/GST} = t_{rx} - RxClkBias$
RxClkDrift	f4	1 ppm	$-2 \cdot 10^{10}$	Receiver clock drift relative to system time (relative frequency error)
TimeSystem	u1		255	Time system of which the offset is provided in this sub-block: 0: GPS time 1: Galileo time 3: GLONASS time
Datum	u1		255	This field defines in which datum the coordinates are expressed: 0: WGS84/ITRS 19: Datum equal to that used by the DGNSS/RTK base station 30: ETRS89 (ETRF2000 realization) 31: NAD83(2011), North American Datum (2011) 32: NAD83(PA11), North American Datum, Pacific plate (2011) 33: NAD83(MA11), North American Datum, Marianas plate (2011) 34: GDA94(2010), Geocentric Datum of Australia (2010) 250: First user-defined datum 251: Second user-defined datum
NrSV	u1		255	Total number of satellites used in the PVT computation.
WACorrInfo	u1		0	Bit field providing information about which wide area corrections have been applied: Bit 0: set if orbit and satellite clock correction information is used Bit 1: set if range correction information is used Bit 2: set if ionospheric information is used Bit 3: set if orbit accuracy information is used (UERE/SISA) Bit 4: set if DO229 Precision Approach mode is active Bits 5-7: Reserved
ReferenceID	u2		65535	This field indicates the reference ID of the differential information used. In case of DGPS or RTK operation, this field is to be interpreted as the base station identifier. In SBAS operation, this field is to be interpreted as the PRN of the geostationary satellite used (from 120 to 158). If multiple base stations or multiple geostationary satellites are used the value is set to 65534.
MeanCorrAge	u2	0.01 s	65535	In case of DGPS or RTK, this field is the mean age of the differential corrections. In case of SBAS operation, this field is the mean age of the 'fast corrections' provided by the SBAS satellites.

SignalInfo	u4		0	Bit field indicating the type of GNSS signals having been used in the PVT computations. If a bit i is set, the signal type having index i has been used. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of <code>SignalInfo</code> .
AlertFlag	u1		0	<p>Bit field indicating integrity related information:</p> <p>Bits 0-1: RAIM integrity flag:</p> <ul style="list-style-type: none"> 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved <p>Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm)</p> <p>Bit 3: Reserved</p> <p>Bit 4: set if either the horizontal or the vertical 2DRMS accuracy is higher than the horizontal or vertical alert limits set by the <code>setNWALevels</code> command.</p> <p>Bits 5-7: Reserved</p>
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	<p>Bit field containing PPP-related information:</p> <p>Bits 0-11: Age of the last seed, in seconds. The age is clipped to 4091s. This field must be ignored when the seed type is 0 (see bits 13-15 below).</p> <p>Bit 12: Reserved</p> <p>Bits 13-15: Type of last seed:</p> <ul style="list-style-type: none"> 0: Not seeded or not in PPP positioning mode 1: Manual seed 2: Seeded from DGPS 3: Seeded from RTKFixed
Padding	u1[..]			Padding bytes, see 3.2.5

Rev 1

ExtEventPVTGeodetic	Number: 4038 "OnChange" interval: each time an external event is detected
---------------------	--

This block contains the position, velocity and time (PVT) solution applicable at the time of an external event, in an ellipsoidal coordinate system.

This block has the same structure and description as the `PVTGeodetic` block, except that the `TOW` and `WNc` fields refer to the time at which the electrical transition on the event pin has been detected (with a millisecond resolution), and that the position is computed at the event time, taking into account a possible user-defined delay set by the `setEventParameters` command.

A user needing the sub-millisecond part of the event time must refer to the `Offset` field of the corresponding `ExtEvent` block. The corresponding `ExtEvent` block is the last of the `ExtEvent` blocks having been output by the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

Mode	u1			<p>Bit field indicating the PVT mode, as follows:</p> <p>Bits 0-3: type of PVT solution:</p> <ul style="list-style-type: none"> 0: No PVT available (the <code>Error</code> field indicates the cause of the absence of the PVT solution) 1: Stand-Alone PVT 2: Differential PVT 3: Fixed location 4: RTK with fixed ambiguities 5: RTK with float ambiguities 6: SBAS aided PVT 7: moving-base RTK with fixed ambiguities 8: moving-base RTK with float ambiguities 10: Precise Point Positioning (PPP) <p>Bits 4-5: Reserved</p> <p>Bit 6: Set if the user has entered the command <code>setPVTMode, base, auto</code> and the receiver is still in the process of determining its fixed position.</p> <p>Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).</p>
Error	u1			<p>PVT error code. The following values are defined:</p> <ul style="list-style-type: none"> 0: No Error 1: Not enough measurements 2: Not enough ephemerides available 3: DOP too large (larger than 15) 4: Sum of squared residuals too large 5: No convergence 6: Not enough measurements after outlier rejection 7: Position output prohibited due to export laws 8: Not enough differential corrections available 9: Base station coordinates unavailable 10: Ambiguities not fixed and user requested to only output RTK-fixed positions <p>Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.</p>
Latitude	f8	1 rad	$-2 \cdot 10^{10}$	Marker latitude, from $-\pi/2$ to $+\pi/2$, positive North of Equator
Longitude	f8	1 rad	$-2 \cdot 10^{10}$	Marker longitude, from $-\pi$ to $+\pi$, positive East of Greenwich
Height	f8	1 m	$-2 \cdot 10^{10}$	Marker ellipsoidal height (with respect to the ellipsoid specified by <code>Datum</code>)
Undulation	f4	1 m	$-2 \cdot 10^{10}$	Geoid undulation computed from the global geoid model defined in the document 'Technical Characteristics of the NAVSTAR GPS, NATO, June 1991'
Vn	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the North direction
Ve	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the East direction
Vu	f4	1 m / s	$-2 \cdot 10^{10}$	Velocity in the 'Up' direction

COG	f4	1 degree	$-2 \cdot 10^{10}$	Course over ground: this is defined as the angle of the vehicle with respect to the local level North, ranging from 0 to 360, and increasing towards east. Set to the do-not-use value when the speed is lower than 0.1m/s.
RxClkBias	f8	1 ms	$-2 \cdot 10^{10}$	Receiver clock bias relative to system time reported in the <code>TimeSystem</code> field. To transfer the receiver time to the system time, use: $t_{GPS/GST} = t_{rx} - RxClkBias$
RxClkDrift	f4	1 ppm	$-2 \cdot 10^{10}$	Receiver clock drift relative to system time (relative frequency error)
TimeSystem	u1		255	Time system of which the offset is provided in this sub-block: 0: GPS time 1: Galileo time 3: GLONASS time
Datum	u1		255	This field defines in which datum the coordinates are expressed: 0: WGS84/ITRS 19: Datum equal to that used by the DGNSS/RTK base station 30: ETRS89 (ETRF2000 realization) 31: NAD83(2011), North American Datum (2011) 32: NAD83(PA11), North American Datum, Pacific plate (2011) 33: NAD83(MA11), North American Datum, Marianas plate (2011) 34: GDA94(2010), Geocentric Datum of Australia (2010) 250: First user-defined datum 251: Second user-defined datum
NrSV	u1		255	Total number of satellites used in the PVT computation.
WACorrInfo	u1		0	Bit field providing information about which wide area corrections have been applied: Bit 0: set if orbit and satellite clock correction information is used Bit 1: set if range correction information is used Bit 2: set if ionospheric information is used Bit 3: set if orbit accuracy information is used (UERE/SISA) Bit 4: set if DO229 Precision Approach mode is active Bits 5-7: Reserved
ReferenceID	u2		65535	This field indicates the reference ID of the differential information used. In case of DGPS or RTK operation, this field is to be interpreted as the base station identifier. In SBAS operation, this field is to be interpreted as the PRN of the geostationary satellite used (from 120 to 158). If multiple base stations or multiple geostationary satellites are used the value is set to 65534.
MeanCorrAge	u2	0.01 s	65535	In case of DGPS or RTK, this field is the mean age of the differential corrections. In case of SBAS operation, this field is the mean age of the 'fast corrections' provided by the SBAS satellites.

SignalInfo	u4		0	Bit field indicating the type of GNSS signals having been used in the PVT computations. If a bit i is set, the signal type having index i has been used. The signal numbers are listed in section 3.2.10. Bit 0 (GPS-C/A) is the LSB of <code>SignalInfo</code> .
AlertFlag	u1		0	<p>Bit field indicating integrity related information:</p> <p>Bits 0-1: RAIM integrity flag:</p> <ul style="list-style-type: none"> 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved <p>Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm)</p> <p>Bit 3: Reserved</p> <p>Bit 4: set if either the horizontal or the vertical 2DRMS accuracy is higher than the horizontal or vertical alert limits set by the <code>setNWALevels</code> command.</p> <p>Bits 5-7: Reserved</p>
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	<p>Bit field containing PPP-related information:</p> <p>Bits 0-11: Age of the last seed, in seconds. The age is clipped to 4091s. This field must be ignored when the seed type is 0 (see bits 13-15 below).</p> <p>Bit 12: Reserved</p> <p>Bits 13-15: Type of last seed:</p> <ul style="list-style-type: none"> 0: Not seeded or not in PPP positioning mode 1: Manual seed 2: Seeded from DGPS 3: Seeded from RTKFixed
Padding	u1[..]			Padding bytes, see 3.2.5

Rev 1

3.3.13 Differential Correction Blocks

DiffCorrIn	Number: 5919
	"OnChange" interval: each time a RTCM or CMR message is received

The `DiffCorrIn` block contains incoming RTCM or CMR messages. The length of the block depends on the message type and contents.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Mode	u1			0: RTCMv2 1: CMRv2 2: RTCMv3 3: RTCMV (a proprietary variant of RTCM2)

Source	u1			<p>Indicates the receiver connection from which the message has been received:</p> <ul style="list-style-type: none"> 0: COM1 1: COM2 2: COM3 3: COM4 4: USB1 5: USB2 6: IP connection 7: SBF file 8: L-Band (message decoded by the built-in L-band demodulator) 9: NTRIP 12: Bluetooth 15: UHF modem 16: IPR connection
If the Mode field is 0 then this field is available:				
RTCM2Words	u4[N]			<p>30-bit words of the RTCM2 message. The Data Word Length (number of 32 bit words) is variable and depends on the RTCM2 message contents. It can be computed by the following piece of C code:</p> <pre>N = 2 + ((RTCM2Words[1]»9) & 0x1f);</pre> <p>N can range from 2 to 33. The first two words are the RTCM2 message header and they are always present.</p> <p>Each of the words is organized as follows:</p> <p>Bits 0-5: 6 parity bits. They are provided for the sake of completeness. Parity doesn't need to be checked, since the DiffCorrIn block only contains valid words.</p> <p>Bits 6-29: 24 information-containing bits of the word. The first received bit is the MSB.</p> <p>Bits 30-31: bit 0 and 1 of the preceding word</p>
If the Mode field is 1 then this field is available:				
CMRMessage	u1[N]			N depends on the CMR message type.
If the Mode field is 2 then this field is available:				
RTCM3Message	u1[N]			N depends on the RTCM 3 message type.
If the Mode field is 3 then this field is available:				
RTCMVMessage	u1[N]			N depends on the RTCMV message type.
Padding	u1[..]			Padding bytes, see 3.2.5

BaseStation	Number:	5949
	"OnChange" interval:	block generated each time a differential correction message related to the base station coordinates is received

The `BaseStation` block contains the ECEF coordinates of the base station the receiver is currently connected to. This block helps users accessing the base station coordinates via SBF instead of having to decode the specific differential correction message (see the `DiffCorrIn` SBF block above).

The interpretation to give to the X, Y, Z ECEF coordinates is dependent on the value of the `Source` field:

Value of Source	Interpretation of X, Y, Z
0, 4 or 10	Coordinate of the L1 phase center
2 or 8	Antenna reference point
9	Proprietary

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
BaseStationID	u2			The base station ID
BaseType	u1			Base station type: 0: Fixed 1: Moving (reserved for future use) 255: Unknown
Source	u1			Source of the base station coordinates: 0: RTCM 2.x (Msg 3) 2: RTCM 2.x (Msg 24) 4: CMR 2.x (Msg 1) 8: RTCM 3.x (Msg 1005 or 1006) 9: RTCMV (Msg 3) 10: CMR+ (Type 2)
Datum	u1		255	Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software
X	f8	1 m		Antenna X coordinate expressed in the datum specified by the <code>Datum</code> field
Y	f8	1 m		Antenna Y coordinate

Z	f8	1 m		Antenna Z coordinate
Padding	u1[..]			Padding bytes, see 3.2.5

RTCMDatum	<p>Number: 4049</p> <p>"OnChange" interval: block generated each time a set of transformation parameters is received</p>
-----------	--

This block reports the source and target datum names as transmitted in RTCM 3.x message types 1021 or 1022. It also reports the corresponding height and quality indicators.

If a service provider only sends out message types 1021 or 1022, this block is transmitted immediately after reception of MT1021 or MT1022. If message types 1023 or 1024 are also sent out, this block is transmitted after the reception of these messages and the `QualityInd` field is set accordingly.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
SourceCRS	c1[32]			Name of the source Coordinate Reference System, right-padded with zeros.
TargetCRS	c1[32]			Name of the target Coordinate Reference System, right-padded with zeros.
Datum	u1			See the <code>Datum</code> field in the <code>PosLocal</code> and <code>PosProjected</code> SBF blocks. Datum is set to 255 if this <code>SourceCRS/TargetCRS</code> pair is currently not used by the receiver.
HeightType	u1			Height Indicator field from MT1021 and MT1022. This field indicates how to interpret the marker height reported in the <code>PosLocal</code> and the <code>PosProjected</code> SBF blocks: 0: Geometrical height 1: Physical height (height definition in target CRS) 2: Physical height (height definition in source CRS)

QualityInd	u1			<p>Bit field indicating the maximum approximation error after applying the transformation:</p> <p>Bits 0-3: horizontal quality indicator:</p> <ul style="list-style-type: none"> 0: Unknown quality 1: Quality better than 21 mm (from MT1021/1022) 2: Quality 21 to 50 mm (from MT1021/1022) 3: Quality 51 to 200 mm (from MT1021/1022) 4: Quality 201 to 500 mm (from MT1021/1022) 5: Quality 501 to 2000 mm (from MT1021/1022) 6: Quality 2001 to 5000 mm (from MT1021/1022) 7: Quality worse than 5001 mm (from MT1021/1022) 9: Quality 0 to 10 mm (from MT1023/1024) 10: Quality 11 to 20 mm (from MT1023/1024) 11: Quality 21 to 50 mm (from MT1023/1024) 12: Quality 51 to 100 mm (from MT1023/1024) 13: Quality 101 to 200 mm (from MT1023/1024) 14: Quality 201 to 500 mm (from MT1023/1024) 15: Quality worse than 501 mm (from MT1023/1024) <p>Bits 4-7: vertical quality indicator, same definition as bits 0-3.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.14 L-Band Demodulator Blocks

LBandTrackerStatus	Number: 4201
	"OnChange" interval: 1s

The `LBandTrackerStatus` block provides general information on the tracking status of the L-band signals.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of L-band trackers for which data is provided in this SBF block, i.e. number of <code>TrackData</code> sub-blocks.
SBLength	u1	1 byte		Length of one sub-block
TrackData		A succession of <i>N</i> <code>TrackData</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

TrackData sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Frequency	u4	1 Hz	0	Nominal frequency of the beam for which data is provided in this sub-block.
Baudrate	u2	1 baud	0	Baudrate of the beam
ServiceID	u2			Service ID of the beam. Set to 0 for the LBAS1 beam. This field must be ignored if the <code>Status</code> field is set to anything else than 3 (Locked).
FreqOffset	f4	1 Hz	$-2 \cdot 10^{10}$	Frequency offset of the demodulator, if available
CN0	u2	0.01 dB-Hz	0	Current C/N_0 value
AvgPower	i2	0.01 dB	-32768	Not applicable, always set to do-not-use value.
AGCGain	i1	1 dB	-128	Not applicable, always set to do-not-use value.
Mode	u1			Current operation mode: 0: normal
Status	u1			Current status: 0: Idle 1: Search 2: FrameSearch 3: Locked
SVID	u1			Satellite ID, see 3.2.9
LockTime	u2	1 s		Lock time to the L-band signal, clipped to 65535 seconds.
Padding	u1[..]			Padding bytes, see 3.2.5

Rev 2 |

Rev 1 |

LBAS1DecoderStatus	<p>Number: 4202</p> <p>"OnChange" interval: Block generated each time a status update is received from the LBAS1 decoder</p>
--------------------	--

The `LBAS1DecoderStatus` block provides general information on the LBAS1 L-band decoder.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Reserved	u1[2]			Reserved for future use. to be ignored by decoding software
Status	u1			Status of the Decoder: 0: No Signal (more than 30 seconds no valid data) 1: Search 2: Locked

Access	u1			Access status: 0: Access disabled 1: Access enabled
GeoGatingMode	u1			GeoGating mode 0: Undefined 1: Non-maritime usage 2: Maritime and non-maritime usage
GeoGatingStatus	u1			GeoGating status. Proprietary information.
Event	u4			Bit field indicating whether an event occurred previously. If this field is not equal to zero, at least one event has occurred. Bit 0: Beamtable Update Bit 1: Station List Update Bit 2: Access Changed Bit 3: Message Received Bits 4-31: Reserved
LeaseTime	u4	1 s	4294967295	Allocated lease time
LeaseRemaining	u4	1 s	4294967295	Remaining lease time
LocalAreaLat	i4	1.0/3600.0 degrees	−2147483648	Local area center latitude, positive North.
LocalAreaLon	i4	1.0/3600.0 degrees	−2147483648	Local area center longitude, positive East of Greenwich.
LocalAreaRadius	u2	1000 m	65535	Local area radius.
LocalAreaStatus	u1			Local area status: 0: Local area disabled 1: Waiting for position 16: Range check 129: User is in range 130: User is out of range 255: Position is too old
Reserved1	u1			Reserved for future use, to be ignored by decoding software.
SubscrEndYear	i1	1 year	−128	Subscription end date, year (2 digits). From 0 to 99.
SubscrEndMonth	i1	1 month	−128	Subscription end date, month. From 1 to 12.
SubscrEndDay	i1	1 day	−128	Subscription end date, day. From 1 to 31.
SubscrEndHour	i1	1 hour	−128	Subscription end date, hour (UTC). From 0 to 23.
PAC	c1[20]			Product activation code, right padded with zeros.
Padding	u1[..]			Padding bytes, see 3.2.5

Rev 1

LBAS1Messages	Number:	4203
	"OnChange" interval:	Block generated each time an over-the-air message is received by the LBAS1 decoder

The `LBAS1Messages` block contains the over-the-air message decoded from LBAS1.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
MessageLength	u2	1 byte		Length of the message in this block. Maximum message length is 512 bytes

Message	c1[MessageLength]			Over-The-Air message
Padding	u1[..]			Padding bytes, see 3.2.5

LBandBeams	Number: 4204 "OnChange" interval: Block generated each time beam status data is received from the LBAS1 decoder
------------	--

This block contains the name, longitude and beam frequency of the L-band geostationary satellites known by the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of L-band beams for which data is provided in this SBF block, i.e. number of <code>BeamInfo</code> sub-blocks.
SBLength	u1	1 byte		Length of one sub-block
BeamInfo		A succession of N <code>BeamInfo</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

BeamInfo sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			SVID associated to the satellite for which information is provided in this sub-block. SVID ranges from 107 to 119. See also section 3.2.9.
SatName	c1[9]			Satellite Name, right padded with zeros
SatLongitude	i2	0.01 degrees	−32768	Satellite Longitude (positive east of Greenwich)
BeamFreq	u4	1 Hz	0	L-band beam center frequency
Padding	u1[..]			Padding bytes, see 3.2.5

3.3.15 Status Blocks

ChannelStatus	Number: 4013
	"OnChange" interval: 10 ms

This block describes the current satellite allocation and tracking status of the active receiver channels. Active channels are channels to which a satellite has been allocated.

This block uses a two-level sub-block structure analogous to that of the `MeasEpoch` block. For each active channel, a `ChannelSatInfo` sub-block contains all satellite-dependent information such as health, azimuth and elevation. Each of these sub-blocks contains `N2` `ChannelStateInfo` sub-blocks, `N2` being the number of active antennas in a given channel (for single-antenna receivers, `N2` is one). The `ChannelStateInfo` reports information such as the tracking status and PVT usage of a given signal type tracked on a given antenna.

Inactive channels are not contained in the `ChannelStatus` block.

Health, tracking and PVT status fields are available for each satellite. These status fields consist of a sequence of up to 8 two-bit fields. Each 2-bit field contains the status of one of the signals transmitted by the satellite. The position of the 2 bits corresponding to a given signal is dependent on the constellation, but is otherwise fixed. It is indicated in the tables below.

GPS:

Reserved		Reserved		Reserved		L5		L2C		P2(Y)		P1(Y)		L1CA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

GLONASS:

Reserved		Reserved		Reserved		L3		L2CA		L2P		Reserved		L1CA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Galileo:

Reserved		E5-AltBOC		E5b		E5a		E6BC		E6A		L1BC		L1A	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

SBAS:

Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		L5		L1	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

COMPASS/BEIDOU:

Reserved		Reserved		Reserved		Reserved		Reserved		B3		B2		B1	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

QZSS:

Reserved		Reserved		Reserved		Reserved		Reserved		L5		L2C		L1CA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

IRNSS:

Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		L5	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of channels for which status are provided in this SBF block, i.e. number of <code>ChannelSatInfo</code> sub-blocks. If <code>N</code> is 0, there are no active channels available for this epoch.
SB1Length	u1	1 byte		Length of a <code>ChannelSatInfo</code> sub-block, excluding the nested <code>ChannelStateInfo</code> sub-blocks
SB2Length	u1	1 byte		Length of a <code>ChannelStateInfo</code> sub-block
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
SatInfo		A succession of <code>N</code> <code>ChannelSatInfo</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

ChannelSatInfo sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			Satellite ID, see 3.2.9
FreqNr	u1		0	For GLONASS satellites, this is the frequency number, with an offset of 8. It ranges from 1 (corresponding to an actual frequency number of -7) to 21 (corresponding to an actual frequency number of 13). For non-GLONASS satellites, FreqNr is reserved and must be ignored by the decoding software.
Reserved1	u1[2]			Reserved for future use, to be ignored by decoding software
Azimuth/RiseSet	u2	1 degree	511 3	bit field: Bits 0-8: Azimuth [0,359]. 0 is North, and Azimuth increases towards East. Bits 9-13: Reserved Bits 14-15: Rise/Set Indicator: 0: Satellite setting 1: Satellite rising 3: Elevation rate unknown
HealthStatus	u2			Sequence of 2-bit health status fields, each of them taking one of the following values: 0 : health unknown, or not applicable 1 : healthy 3 : unhealthy The 2-bit health status is a condensed version of the health status as sent by the satellite. For SBAS, the health status is set from the almanac data (MT17).
Elevation	i1	1 degree	-128	Elevation [-90,90] relative to local horizontal plane
N2	u1			Number of ChannelStateInfo blocks following this ChannelSatInfo block. There is one ChannelStateInfo sub-block per antenna.
RxChannel	u1			Channel number, see section 3.2.11.
Reserved2	u1			Reserved for future use, to be ignored by decoding software
Padding	u1[...]			Padding bytes, see 3.2.5
StateInfo		A succession of N2 ChannelStateInfo sub-blocks, see definition below

ChannelStateInfo sub-block definition:

Parameter	Type	Units & Scale Factor	Description
Antenna	u1		Antenna number (0 for main antenna)
Reserved	u1		Reserved for future use, to be ignored by decoding software
TrackingStatus	u2		Sequence of 2-bit tracking status fields, each of them taking one of the following values: 0: idle or not applicable 1: Search 2: Sync 3: Tracking
PVTStatus	u2		Sequence of 2-bit PVT status fields, each of them taking one of the following values: 0: not used 1: waiting for ephemeris 2: used 3: rejected
PVTInfo	u2		Internal info
Padding	u1[..]		Padding bytes, see 3.2.5

ReceiverStatus	Number: 4014 "OnChange" interval: 1s
----------------	---

The `ReceiverStatus` block provides general information on the status of the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
CPUload	u1	1 %	255	Load on the receiver's CPU. The load should stay below 80% in normal operation. Higher loads might result in data loss.

ExtError	u1			<p>Bit field reporting external errors, i.e. errors detected in external data. Upon detection of an error, the corresponding bit is set for a duration of one second, and then resets.</p> <p>Bit 0: SISERROR: set if a violation of the signal-in-space ICD has been detected for at least one satellite while that satellite is reported as healthy. Use the command "lif, SisError" for details.</p> <p>Bit 1: DIFFCORRERROR: set when an anomaly has been detected in an incoming differential correction stream, causing the receiver to fail to decode the corrections. Use the command "lif, DiffCorrError" for details.</p> <p>Bit 2: EXTSENSORERROR: set when a malfunction has been detected on at least one of the external sensors connected to the receiver. Use the command "lif, ExtSensorError" for details.</p> <p>Bit 3: SETUPERROR: set when a configuration/setup error has been detected. An example of such error is when a remote NTRIP Caster is not reachable. Use the command "lif, SetupError" for details.</p> <p>Bits 4-7: Reserved</p>
UpTime	u4	1 s		<p>Number of seconds elapsed since the start-up of the receiver, or since the last reset.</p>

RxState	u4			<p>Bit field indicating the status of key components of the receiver:</p> <p>Bit 0: Reserved</p> <p>Bit 1: Reserved</p> <p>Bit 2: EXT_REF: in PolaRxS receivers, this bit is set to indicate correct operation of the internal OCXO reference.</p> <p>Bit 3: PPS_IN: this bit is set if a pulse has been detected on the 1PPS input connector and the receiver time has been synchronized with this pulse.</p> <p>Bit 4: WNSET: see corresponding bit in the SyncLevel field of the ReceiverTime block.</p> <p>Bit 5: TOWSET: see corresponding bit in the SyncLevel field of the ReceiverTime block.</p> <p>Bit 6: FINETIME: see corresponding bit in the SyncLevel field of the ReceiverTime block.</p> <p>Bit 7: DISK_ACTIVITY: this bit is set for one second each time data is logged to the internal disk (DSK1). If the logging rate is larger than 1 Hz, set continuously.</p> <p>Bit 8: DISK_FULL: this bit is set when the internal disk (DSK1) is full. A disk is full when it is filled to 95% of its total capacity.</p> <p>Bit 9: DISK_MOUNTED: this bit is set when the internal disk (DSK1) is mounted.</p> <p>Bit 10: INT_ANT: this bit is set when the RF signal is taken from the internal antenna input, and cleared when it comes from the external antenna input (only applicable on receiver models featuring an internal antenna input).</p> <p>Bit 11: REFOUT_LOCKED: if set, the 10-MHz frequency provided at the REF OUT connector is locked to GNSS time. Otherwise it is free-running.</p> <p>Bits 12-31: Reserved</p>
---------	----	--	--	---

Rev 1

RxError	u4			<p>Bit field indicating whether an error occurred previously. If this field is not equal to zero, at least one error has been detected.</p> <p>Bit 0: Reserved</p> <p>Bit 1: Reserved</p> <p>Bit 2: Reserved</p> <p>Bit 3: SOFTWARE: set upon detection of a software warning or error. This bit is reset by the command "lif, error".</p> <p>Bit 4: WATCHDOG: set when the watchdog expired at least once since the last power-on.</p> <p>Bit 5: Reserved</p> <p>Bit 6: CONGESTION: set when an output data congestion has been detected on at least one of the communication ports of the receiver during the last second.</p> <p>Bit 7: Reserved</p> <p>Bit 8: MISSEDEVENT: set when an external event congestion has been detected during the last second. It indicates that the receiver is receiving too many events on its EVENTx pins.</p> <p>Bit 9: CPUOVERLOAD: set when the CPU load is larger than 90%. If this bit is set, receiver operation may be unreliable and the user must decrease the processing load by following the recommendations in the Firmware User Manual.</p> <p>Bit 10: INVALIDCONFIG: set if one or more configuration file (e.g. permissions) is invalid or absent.</p> <p>Bit 11: OUTOFGEOFENCE: set if the receiver is currently out of its permitted region of operation (geofencing).</p> <p>Bit 12: Reserved for future use</p> <p>Bit 13: Reserved for future use</p> <p>Bit 14: Reserved for future use</p> <p>Bit 15: Reserved for future use</p> <p>Bit 16: Reserved for future use</p> <p>Bits 17-31: Reserved</p>
N	u1			Number of AGCState sub-blocks this block contains.
SBLength	u1	1 byte		Length of a AGCState sub-block.
CmdCount	u1		0	Command cyclic counter, incremented each time a command is entered that changes the receiver configuration. After the counter has reached 255, it resets to 1.
Temperature	u1	1 °C	0	Not applicable.
AGCState		A succession of N AGCState sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

AGCState sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
FrontEndID	u1			<p>Bit field indicating the frontend code and antenna ID:</p> <p>Bits 0-4: frontend code:</p> <ul style="list-style-type: none"> 0: GPSL1/E1 1: GLOL1 2: E6 3: GPSL2 4: GLOL2 5: L5/E5a 6: E5b/B2 7: E5(a+b) 8: Combined GPS/GLONASS/SBAS/Galileo L1 9: Combined GPS/GLONASS L2 10: MSS/L-band 11: B1 12: B3 <p>Bits 5-7: antenna ID: 0 for main, 1 for <i>Aux1</i> and 2 for <i>Aux2</i></p>
Gain	i1	1 dB	-128	<p>AGC gain, in dB.</p> <p>The Do-Not-Use value is used to indicate that the frontend PLL is not locked.</p>
SampleVar	u1		0	Normalized variance of the IF samples. The nominal value for this variance is 100.
BlankingStat	u1	1 %		Current percentage of samples being blanked by the pulse blanking unit. This field is always 0 for receiver without pulse blanking unit.
Padding	u1[..]			Padding bytes, see 3.2.5

SatVisibility	Number: 4012
	"OnChange" interval: 1s

This block contains the azimuth and elevation of all the satellites above the horizon for which the ephemeris or almanac is available.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of satellites for which information is provided in this SBF block, i.e. number of <code>SatInfo</code> sub-blocks.
SBLength	u1	1 byte		Length of one <code>SatInfo</code> sub-block
SatInfo		A succession of <i>N</i> <code>SatInfo</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

SatInfo sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
SVID	u1			Satellite ID, see 3.2.9
FreqNr	u1		0	For GLONASS satellites, this is the frequency number, with an offset of 8. It ranges from 1 (corresponding to an actual frequency number of -7) to 21 (corresponding to an actual frequency number of 13). For non-GLONASS satellites, FreqNr is reserved and must be ignored by the decoding software.
Azimuth	u2	0.01 degrees	65535	Azimuth. 0 is North, and azimuth increases towards East.
Elevation	i2	0.01 degrees	−32768	Elevation relative to local horizontal plane.
RiseSet	u1			Rise/set indicator: 0: satellite setting 1: satellite rising 255: elevation rate unknown
SatelliteInfo	u1			Satellite visibility info based on: 1: almanac 2: ephemeris 255: unknown
Padding	u1[..]			Padding bytes, see 3.2.5

InputLink	Number: 4090
	"OnChange" interval: 1s

The `InputLink` block reports statistics of the number of bytes and messages received and accepted on each active connection descriptor.

Per connection descriptor, the receiver maintains two byte counters (`NrBytesReceived` and `NrBytesAccepted`) and two message counters (`NrMsgReceived` and `NrMsgAccepted`), which are reported in the sub-blocks. These counters provide useful information on the quality of the transmission link, and of the bandwidth efficiency.

These counters (as well as the age of the last message) are reset simultaneously on the following events:

- start-up of the receiver
- overflow of one of the counters
- change of input type
- deactivation of a connection descriptor, e.g. on disconnection of USB or IP ports.

There is one sub-block per connection descriptor for which statistics is available.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of connection descriptors for which communication link statistics are included
SBLength	u1	1 byte		Length of one <code>InputStatsSub</code> sub-block.
InputStats		A succession of <i>N</i> <code>InputStatsSub</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

InputStatsSub sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description																																	
CD	u1			Identifier of the connection to which these statistics apply:																																	
				<table><tr><th>Value of CD</th><th>Connection type</th><th>Example</th></tr><tr><td>0-31</td><td>COMx, with $x=CD$</td><td>1: COM1</td></tr><tr><td>32-63</td><td>USBx, with $x=CD-32$</td><td>33: USB1</td></tr><tr><td>64-95</td><td>IPx, with $x=CD-54$</td><td>64:IP10</td></tr><tr><td>96-127</td><td>DSKx, with $x=CD-96$</td><td>97:DSK1</td></tr><tr><td>128-159</td><td>NTRx, with $x=CD-128$ (NTRIP connections)</td><td>129:NTR1</td></tr><tr><td>160-191</td><td>IPsx, with $x=CD-160$ (IP server connections)</td><td>161:IPS1</td></tr><tr><td>192</td><td>BT01 (Bluetooth connection)</td><td></td></tr><tr><td>196</td><td>UHF1 (UHF Modem)</td><td></td></tr><tr><td>200-205</td><td>IPRx, with $x=CD-200$ (IP receive connections)</td><td>201:IPR1</td></tr><tr><td>206-255</td><td>Reserved</td><td></td></tr></table>	Value of CD	Connection type	Example	0-31	COMx, with $x=CD$	1: COM1	32-63	USBx, with $x=CD-32$	33: USB1	64-95	IPx, with $x=CD-54$	64:IP10	96-127	DSKx, with $x=CD-96$	97:DSK1	128-159	NTRx, with $x=CD-128$ (NTRIP connections)	129:NTR1	160-191	IPsx, with $x=CD-160$ (IP server connections)	161:IPS1	192	BT01 (Bluetooth connection)		196	UHF1 (UHF Modem)		200-205	IPRx, with $x=CD-200$ (IP receive connections)	201:IPR1	206-255	Reserved	
				Value of CD	Connection type	Example																															
				0-31	COMx, with $x=CD$	1: COM1																															
				32-63	USBx, with $x=CD-32$	33: USB1																															
				64-95	IPx, with $x=CD-54$	64:IP10																															
				96-127	DSKx, with $x=CD-96$	97:DSK1																															
				128-159	NTRx, with $x=CD-128$ (NTRIP connections)	129:NTR1																															
				160-191	IPsx, with $x=CD-160$ (IP server connections)	161:IPS1																															
				192	BT01 (Bluetooth connection)																																
				196	UHF1 (UHF Modem)																																
				200-205	IPRx, with $x=CD-200$ (IP receive connections)	201:IPR1																															
206-255	Reserved																																				
Type	u1			Type of data: 0: none 1: DaisyChain (includes "echo" messages) 32: CMD 33: SBF 34: AsciiDisplay (see setDataInOut command) 64: NMEA 96: RTCMv2 97: RTCMv3 98: CMRv2 99: RTCMV (a proprietary variant of RTCMv2) 128: MTI (IMU sensor) 129: MMQ (IMU sensor) 130: ELLIPSE (IMU sensor) 160: ASCIIIn																																	
				AgeOfLastMessage	u2	1 s	65535	Age of the last accepted message. If the age is older than 65534s, it is clipped to 65534s.																													
				NrBytesReceived	u4	1 byte		Total number of bytes received ⁽¹⁰⁾																													

⁽¹⁰⁾ Note that, for RTCM 2.x, one 8-bit byte contains 6 RTCM data bits.

NrBytesAccepted	u4	1 byte	4294967295	<p>Total number of bytes ⁽¹⁰⁾ in messages that passed the check for this type of input (CRC, parity check, ...).</p> <p>The ratio of NrBytesAccepted to NrBytesReceived gives an indication of the quality of the communication link.</p>
NrMsgReceived	u4	1 message	4294967295	Total number of messages of type Type received.
NrMsgAccepted	u4	1 message	4294967295	<p>Total number of messages of type Type that were interpreted and used by the receiver.</p> <p>The ratio of NrMsgAccepted to NrMsgReceived gives an indication of the bandwidth usage efficiency</p>
Padding	u1[..]			Padding bytes, see 3.2.5

OutputLink	Number: 4091
	"OnChange" interval: 1s

The `OutputLink` block reports statistics of the number of bytes sent on each active connection descriptor.

Per connection descriptor, the receiver maintains two byte counters `NrBytesProduced` and `NrBytesSent`, which are reported in the sub-block. They provide an indication of the amount of data output and data lost on a given connection.

These counters are reset simultaneously on the following events:

- start-up of the receiver
- overflow of one of the counters
- deactivation of a connection descriptor, e.g. on disconnection of USB or IP ports
- change of COM port settings.

There is one `OutputStatsSub` sub-block per connection descriptor for which statistics is available. Each `OutputStatsSub` sub-block contains a number of `OutputTypeSub` sub-blocks. These sub-blocks indicate which data type has been output through the connection in question during the last second. If no output happened during the last second, there is no `OutputTypeSub` sub-block.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.

WNC	u2	1 week	65535	<p>The GPS week number associated with the <code>TOW</code>. <code>WNC</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks.</p> <p>By definition of the Galileo system time, <code>WNC</code> is also the Galileo week number plus 1024.</p>
N1	u1			Number of <code>OutputStatsSub</code> sub-blocks in this <code>OutputLink</code> block.
SB1Length	u1	1 byte		Length of an <code>OutputStatsSub</code> sub-block, excluding the nested <code>OutputTypeSub</code> sub-block
SB2Length	u1	1 byte		Length of an <code>OutputTypeSub</code> sub-block
Reserved	u1[3]			Reserved for future use
<i>OutputStats</i>		<i>A succession of N1 OutputStatsSub sub-blocks, see definition below</i>
Padding	u1[..]			Padding bytes, see 3.2.5

OutputStatsSub sub-block definition:

Parameter	Type	Units & Scale Factor	Description																																	
CD	u1		Identifier of the connection to which these statistics apply:																																	
			<table><tr><th>Value of CD</th><th>Connection type</th><th>Example</th></tr><tr><td>0-31</td><td>COMx, with $x=CD$</td><td>1: COM1</td></tr><tr><td>32-63</td><td>USBx, with $x=CD-32$</td><td>33: USB1</td></tr><tr><td>64-95</td><td>IPx, with $x=CD-54$</td><td>64:IP10</td></tr><tr><td>96-127</td><td>DSKx, with $x=CD-96$</td><td>97:DSK1</td></tr><tr><td>128-159</td><td>NTRx, with $x=CD-128$ (NTRIP connections)</td><td>129:NTR1</td></tr><tr><td>160-191</td><td>IPsx, with $x=CD-160$ (IP server connections)</td><td>161:IPS1</td></tr><tr><td>192</td><td>BT01 (Bluetooth connection)</td><td></td></tr><tr><td>196</td><td>UHF1 (UHF Modem)</td><td></td></tr><tr><td>200-205</td><td>IPRx, with $x=CD-200$ (IP receive connections)</td><td>201:IPR1</td></tr><tr><td>206-255</td><td>Reserved</td><td></td></tr></table>	Value of CD	Connection type	Example	0-31	COMx, with $x=CD$	1: COM1	32-63	USBx, with $x=CD-32$	33: USB1	64-95	IPx, with $x=CD-54$	64:IP10	96-127	DSKx, with $x=CD-96$	97:DSK1	128-159	NTRx, with $x=CD-128$ (NTRIP connections)	129:NTR1	160-191	IPsx, with $x=CD-160$ (IP server connections)	161:IPS1	192	BT01 (Bluetooth connection)		196	UHF1 (UHF Modem)		200-205	IPRx, with $x=CD-200$ (IP receive connections)	201:IPR1	206-255	Reserved	
			Value of CD	Connection type	Example																															
			0-31	COMx, with $x=CD$	1: COM1																															
			32-63	USBx, with $x=CD-32$	33: USB1																															
			64-95	IPx, with $x=CD-54$	64:IP10																															
			96-127	DSKx, with $x=CD-96$	97:DSK1																															
			128-159	NTRx, with $x=CD-128$ (NTRIP connections)	129:NTR1																															
			160-191	IPsx, with $x=CD-160$ (IP server connections)	161:IPS1																															
			192	BT01 (Bluetooth connection)																																
			196	UHF1 (UHF Modem)																																
			200-205	IPRx, with $x=CD-200$ (IP receive connections)	201:IPR1																															
206-255	Reserved																																			
N2	u1		Number of <code>OutputTypeSub</code> sub-blocks included at the end of this <code>OutputStatsSub</code> sub-block																																	
AllowedRate	u2	1 kbyte / s	Maximum datarate recommended on this connection																																	
NrBytesProduced	u4	1 byte	Total number of bytes produced by the receiver																																	
NrBytesSent	u4	1 byte	Total number of bytes actually sent (i.e. without congestions or transmission errors). The ratio of <code>NrBytesSent</code> to <code>NrBytesProduced</code> gives an indication of the amount of bandwidth overload.																																	
NrClients	u1		Number of clients currently connected to this connection. Most connection types can only serve one client at a time, but each IP server (IPS) port can serve up to eight simultaneous clients. Note that when <code>NrClients</code> is more than one, the fields <code>NrBytesProduced</code> and <code>NrBytesSent</code> are the total number of bytes produced and sent to all client.																																	
Reserved	u1[3]		Reserved for future use																																	
Padding	u1[.]		Padding bytes, see 3.2.5																																	
OutputType	A succession of <code>N2 OutputTypeSub</code> sub-blocks, see definition below																																	

Rev 1

OutputTypeSub sub-block definition:

Parameter	Type	Units & Scale Factor	Description
Type	u1		Type of data: 0: none 1: DaisyChain (includes "echo" messages) 32: CMD 33: SBF 34: AsciiDisplay (see setDataInOut command) 64: NMEA 96: RTCMv2 97: RTCMv3 98: CMRv2 99: RTCMV (a proprietary variant of RTCMv2) 128: MTI (IMU sensor) 129: MMQ (IMU sensor) 130: ELLIPSE (IMU sensor) 160: ASCIIIn
Percentage	u1	1 %	Percentage of the produced bytes that belong to this type (during the last second)
Padding	u1[..]		Padding bytes, see 3.2.5

NTRIPClientStatus	Number: 4053 "OnChange" interval: 1s
-------------------	---

This block reports the current status of the NTRIP client connections.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of NTRIP client connections for which status is provided in this block, i.e. number of <code>NTRIPClientConnection</code> sub-blocks.
SBLength	u1	1 byte		Length of one <code>NTRIPClientConnection</code> sub-block
<i>NTRIPClientConnection</i>		<i>A succession of N NTRIPClientConnection sub-blocks, see definition below</i>
Padding	u1[..]			Padding bytes, see 3.2.5

NTRIPClientConnection sub-block definition:

Parameter	Type	Units & Scale Factor	Description
CDIndex	u1		Index of the NTRIP connection (1 for NTR1, 2 for NTR2, etc) for which status is provided in this sub-block.
Status	u1		NTRIP client status: 0: Connection disabled 1: Initializing 2: Running, differential corrections are being received and the link statistics is available in the <code>InputLink</code> block. 3: Error detected, the error code is provided in the next field. 4: Retrying, client encountered an error, we are retrying to connect. The error code is provided in the next field.
ErrorCode	u1		NTRIP error code: 0: No error 1: Initialization error (e.g. source table retrieval failure) 2: Authentication error 3: Connection error 4: Mountpoint does not exist 5: Waiting for GGA 6: GGA sending disabled when required by mountpoint 7: Resolving host failed 254: Unknown error
Padding	u1[..]		Padding bytes, see 3.2.5

IPStatus	Number: 4058 "OnChange" interval: output each time one or more IP parameters change
----------	--

This block contains the receiver's IP address, the gateway, the netmask and the MAC address.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
MACAddress	u1[6]			MAC address. The first byte corresponds to the MSB of the address.
IPAddress	u1[16]		All elements set to 0	IP address. For future upgradability, this field can contain a 128-bit IPv6 address. In the current firmware version, the first 12 bytes are always set to 0, and the last 4 bytes contain the IPv4 IP address, or are set to zero if the IP address is not known or not applicable.

Gateway	u1[16]		All elements set to 0	Gateway address. For future upgradability, this field can contain a 128-bit IPv6 address. In the current firmware version, the first 12 bytes are always set to 0, and the last 4 bytes contain the IPv4 IP address, or are set to zero if the gateway address is not known or not applicable.
Netmask	u1		255	Number of bits used to identify the network (CIDR notation).
Padding	u1[..]			Padding bytes, see 3.2.5

QualityInd	Number: 4082
	"OnChange" interval: 1s

The `QualityInd` block contains quality indicators for the main functions of the receiver. Each quality indicator is a value from 0 to 10, 0 corresponding to poor quality and 10 to very high quality.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of quality indicators contained in this block
Reserved	u1			Reserved for future use, to be ignored by decoding software.

Indicators	u2[N]			<p>N successive quality indicators, coded as follows:</p> <p>Bits 0-7: Quality indicator type:</p> <ul style="list-style-type: none"> 0: Overall quality 1: GNSS signals from main antenna 2: GNSS signals from aux1 antenna 11: Main antenna cabling 12: Aux1 antenna cabling 21: CPU headroom 30: Base station measurements quality. This indicator is only available in RTK mode. A low value could for example hint at severe multipath or interference at the base station, or also at ionospheric scintillation. <p>Bits 8-11: Value of this quality indicator (from 0 for low quality to 10 for high quality)</p> <p>Bits 12-15: Reserved for future use, to be ignored by decoding software.</p>
Padding	u1[..]			Padding bytes, see 3.2.5

DiskStatus	Number: 4059 "OnChange" interval: 1s
------------	---

This block reports the size and usage of the disks mounted on the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
N	u1			Number of <code>DiskData</code> sub-blocks this block contains.
SBLength	u1	1 byte		Length of one <code>DiskData</code> sub-blocks in bytes.
Reserved	u1[4]			Reserved for future use
DiskData		A succession of <i>N</i> <code>DiskData</code> sub-blocks, see definition below
Padding	u1[...]			Padding bytes, see 3.2.5

DiskData sub-block definition:

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
DiskID	u1			ID of the disk, starting at 1 for the internal SD Memory Card.
Status	u1			<p>Bit field:</p> <p>Bit 0: DISK_MOUNTED: bit set when the disk is mounted.</p> <p>Bit 1: DISK_FULL: bit set when the disk is full. A disk is full when it is filled to 95% of its total capacity.</p> <p>Bit 2: DISK_ACTIVITY: bit set for one second each time data is written to the disk. If the logging rate is larger than 1 Hz, set continuously.</p> <p>Bit 3: LOGGING_ENABLED: bit set when at least one file is open on the disk, regardless of the logging rate.</p> <p>Bits 4-7: Reserved</p>
DiskUsageMSB	u2		65535 ⁽¹¹⁾	16 MSB of the total disk usage. The disk usage in bytes is given by $\text{DiskUsageMSB} \times 4294967296 + \text{DiskUsageLSB}$.
DiskUsageLSB	u4		4294967295 ⁽¹¹⁾	32 LSB of the total disk usage. The disk usage in bytes is given by $\text{DiskUsageMSB} \times 4294967296 + \text{DiskUsageLSB}$.
DiskSize	u4	1 Mbyte	0	Total size of the disk, in megabytes.
CreateDeleteCount	u1			Counter incremented by one each time a file or a folder is created or deleted on this disk. This counter starts at zero at receiver start-up and restarts at zero after having reached 255.
Padding	u1[..]			Padding bytes, see 3.2.5

⁽¹¹⁾ The disk usage is invalid if both DiskUsageMSB is 65535 and DiskUsageLSB is 4294967295.

3.3.16 Miscellaneous Blocks

ReceiverSetup	Number:	5902
	"OnChange" interval:	Block generated each time the user invokes one of the following commands: setAntennaOffset , setMarkerParameters or setObserverParameters

The `ReceiverSetup` block contains parameters related to the receiver set-up. This block provides most of the information to be included in a RINEX header.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Reserved	u1[2]			2 bytes reserved for future use, to be ignored by decoding software
MarkerName	c1[60]			Name of the marker, this is a 60-character string, right padded with zeros.
MarkerNumber	c1[20]			Marker identification, this is a 20-character string, right padded with zeros

Observer	c1[20]			Observer description, this is a 20-character string, right padded with zeros.
Agency	c1[40]			Observer's agency description, this is a 40-character string, right padded with zeros
RxSerialNumber	c1[20]			Receiver serial number, this is a 20-character string, right padded with zeros.
RxName	c1[20]			Receiver core name, this is a 20-character string, right padded with zeros.
RxVersion	c1[20]			Receiver firmware version, this is a 20-character string, right padded with zeros.
AntSerialNbr	c1[20]			Serial number of the main antenna, this is a 20-character string, right padded with zeros.
AntType	c1[20]			Type of the main antenna, this is a 20-character string, right padded with zeros
deltaH	f4	1 m		δH offset of the main antenna
deltaE	f4	1 m		δE offset of the main antenna
deltaN	f4	1 m		δN offset of the main antenna
MarkerType	c1[20]			Marker type, this is a 20-character string, right padded with zeros
GNSSFirmwareVersion	c1[40]			Version tag of the GNSS firmware installed on the receiver. This is a 40-character string, right padded with zeros.
ProductName	c1[40]			Product name. This is a 40-character string, right padded with zeros.
Padding	u1[.]			Padding bytes, see 3.2.5

Rev 1

Rev 2

Rev 3

Commands	Number: 4015 "OnChange" interval: each time a user command is entered
----------	--

Every time the user sends a command, a `Commands` block is output on all ports for which this block is enabled. The `Commands` SBF block is inserted in the SBF stream at the very moment when the command starts to take effect.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software.
CmdData	u1[N]			Command data, this is the command in the SNMP' format (reserved for maintenance and support only).
Padding	u1[..]			Padding bytes, see 3.2.5

Comment	Number: 5936 "OnChange" interval: block generated each time a comment is entered with setObserverComment
---------	--

The **Comment** block contains a comment string as entered with the **setObserverComment** command.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW . WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
CommentLn	u2			Length of the Comment string, in characters. The maximum length of a comment is 120 characters.

Comment	c1[CommentLn]			Comment string, as entered with the setObserverComment command. Note that this string is not terminated by the "\0" character.
Padding	u1[..]			Padding bytes, see 3.2.5

BBSamples	Number: 4040
	"OnChange" interval: block generated each time new baseband samples are ready (typically at 2Hz)

The **BBSamples** block contains a series of successive complex baseband samples. These samples can be used for signal monitoring and for spectral analysis of the GNSS bands supported by the receiver.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The Sync1 field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The Sync2 field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The CRC field is the 16-bit CRC of all the bytes in an SBF block from and including the ID field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The ID field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The Length field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the TOW . WNc is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, WNc is also the Galileo week number plus 1024.
N	u2			Number of complex baseband samples contained in this block
Info	u1			Bit field as follows: Bits 0-2: Antenna ID: antenna from which the samples have been taken: 0 for main, 1 for <i>Aux1</i> and 2 for <i>Aux2</i> . Bits 3-7: Reserved
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software.
SampleFreq	u4	1 Hz		Sampling frequency in Hz.

LOFreq	u4	1 Hz		Frequency of the local oscillator (LO) used to down-convert the RF signal to baseband.
Samples	u2[N]			N successive complex baseband samples (I+jQ), coded as follows: Bits 0-7: 8-bit Q component, two's complement. Bits 8-15: 8-bit I component, two's complement.
Padding	u1[..]			Padding bytes, see 3.2.5

ASCIIN	Number: 4075
	"OnChange" interval: block generated each time an ASCII string is received

The ASCIIN block contains a string that has been received on one of the receiver's connection ports.

More specifically, this block is output each time an end-of-line character is received on a communication port configured to receive ASCIIN input (with the **setDataInOut** command). The string reported in this block contains all characters received since the previous occurrence of an end-of-line character.

The maximum length of the string is 2000 characters. If there are more than 2000 characters between the occurrence of two successive end-of-line characters, the string is discarded

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.

CD	u1			Identifier of the connection to which these statistics apply:																																	
				<table><tr><th>Value of CD</th><th>Connection type</th><th>Example</th></tr><tr><td>0-31</td><td>COMx, with $x=CD$</td><td>1: COM1</td></tr><tr><td>32-63</td><td>USBx, with $x=CD-32$</td><td>33: USB1</td></tr><tr><td>64-95</td><td>IPx, with $x=CD-54$</td><td>64:IP10</td></tr><tr><td>96-127</td><td>DSKx, with $x=CD-96$</td><td>97:DSK1</td></tr><tr><td>128-159</td><td>NTRx, with $x=CD-128$ (NTRIP connections)</td><td>129:NTR1</td></tr><tr><td>160-191</td><td>IPSx, with $x=CD-160$ (IP server connections)</td><td>161:IPS1</td></tr><tr><td>192</td><td>BT01 (Bluetooth connection)</td><td></td></tr><tr><td>196</td><td>UHF1 (UHF Modem)</td><td></td></tr><tr><td>200-205</td><td>IPRx, with $x=CD-200$ (IP receive connections)</td><td>201:IPR1</td></tr><tr><td>206-255</td><td>Reserved</td><td></td></tr></table>	Value of CD	Connection type	Example	0-31	COM x , with $x=CD$	1: COM1	32-63	USB x , with $x=CD-32$	33: USB1	64-95	IP x , with $x=CD-54$	64:IP10	96-127	DSK x , with $x=CD-96$	97:DSK1	128-159	NTR x , with $x=CD-128$ (NTRIP connections)	129:NTR1	160-191	IPS x , with $x=CD-160$ (IP server connections)	161:IPS1	192	BT01 (Bluetooth connection)		196	UHF1 (UHF Modem)		200-205	IPR x , with $x=CD-200$ (IP receive connections)	201:IPR1	206-255	Reserved	
				Value of CD	Connection type	Example																															
				0-31	COM x , with $x=CD$	1: COM1																															
				32-63	USB x , with $x=CD-32$	33: USB1																															
				64-95	IP x , with $x=CD-54$	64:IP10																															
				96-127	DSK x , with $x=CD-96$	97:DSK1																															
				128-159	NTR x , with $x=CD-128$ (NTRIP connections)	129:NTR1																															
				160-191	IPS x , with $x=CD-160$ (IP server connections)	161:IPS1																															
				192	BT01 (Bluetooth connection)																																
				196	UHF1 (UHF Modem)																																
200-205	IPR x , with $x=CD-200$ (IP receive connections)	201:IPR1																																			
206-255	Reserved																																				
Reserved1	u1[3]			Reserved, contents to be ignored.																																	
StringLength	u2			Length of ASCIIString in characters.																																	
SensorModel	c1[20]			Not supported, reserved for future use.																																	
SensorType	c1[20]			Not supported, reserved for future use.																																	
Reserved2	u1[20]			Reserved, contents to be ignored.																																	
ASCIIString	c1[StringLn]			ASCII string. Note that this string is not terminated by the "\0" character. The string does not include the end-of-line character(s) (carrier return and/or line feed).																																	
Padding	u1[..]			Padding bytes, see 3.2.5																																	

3.3.17 Deprecated or Obsolete Blocks

BaseLine	Number: 5950 "OnChange" interval: 10 ms
----------	--

The `BaseLine` block contains the relative position of the receiver with respect to the base station in case of DGPS or RTK positioning.



This block is deprecated and should not be used in new designs. Use the `BaseVectorGeod` block instead.

Parameter	Type	Units & Scale Factor	Do-Not-Use Value	Description
Sync1	c1			The <code>Sync1</code> field is the first byte of a 2-byte array. It is always set to 0x24 (decimal 36, ASCII '\$'). Together with the second byte it identifies the beginning of any SBF block and can be used for synchronization.
Sync2	c1			The <code>Sync2</code> field is the second byte of a 2-byte array. It is always set to 0x40 (decimal 64, ASCII '@'). Together with the first byte it identifies the beginning of any SBF block and can be used for synchronization.
CRC	u2			The <code>CRC</code> field is the 16-bit CRC of all the bytes in an SBF block from and including the <code>ID</code> field to the last byte of the block. The generator polynomial for this CRC is the so-called CRC-CCITT polynomial: $x^{16} + x^{12} + x^5 + x^0$. The CRC is computed in the forward direction using a seed of 0, no reverse and no final XOR.
ID	u2			The <code>ID</code> field is a 2-byte block ID, which uniquely identifies the block type and its contents. It is a bit field with the following definition: Bits 0-12: Block number Bits 13-15: Block revision number, starting from 0 at the initial block definition, and incrementing each time backwards-compatible changes are performed to the block (see section 3.2.6)
Length	u2	1 byte		The <code>Length</code> field is a 2-byte unsigned integer containing the size of the SBF block. It is the total number of bytes in the SBF block including the header. It is always a multiple of 4.
TOW	u4	0.001 s	4294967295	Time-Of-Week : Time-tag, expressed in whole milliseconds from the beginning of the current Galileo/GPS week.
WNc	u2	1 week	65535	The GPS week number associated with the <code>TOW</code> . <code>WNc</code> is a continuous week count (hence the "c"). It is not affected by GPS week rollovers, which occur every 1024 weeks. By definition of the Galileo system time, <code>WNc</code> is also the Galileo week number plus 1024.
BaseStationID	u2		65535	The base station ID
East	f8	1 m	$-2 \cdot 10^{10}$	East baseline component
North	f8	1 m	$-2 \cdot 10^{10}$	North baseline component
Up	f8	1 m	$-2 \cdot 10^{10}$	Up baseline component
Padding	u1[..]			Padding bytes, see 3.2.5

Chapter 4

Index of Commands

A

AGCMode

setAGCMode, getAGCMode
sam, gam, 89

AntennaInfo

lstAntennaInfo
lai, 70

AntennaLocation

setAntennaLocation, getAntennaLocation
sal, gal, 102

AntennaOffset

setAntennaOffset, getAntennaOffset
sao, gao, 103

AttitudeOffset

setAttitudeOffset, getAttitudeOffset
sto, gto, 104

C

ChannelAllocation

setChannelAllocation, getChannelAllocation
sca, gca, 90

ChannelConfiguration

getChannelConfiguration
gcc, 91

ClockSyncThreshold

setClockSyncThreshold, getClockSyncThreshold
scst, gcst, 138

CMRv2Formatting

setCMRv2Formatting, getCMRv2Formatting
sc2f, gc2f, 191

CMRv2Interval

setCMRv2Interval, getCMRv2Interval
sc2i, gc2i, 192

CMRv2Message2

setCMRv2Message2, getCMRv2Message2
sc2m, gc2m, 193

CMRv2Output

- setCMRv2Output, getCMRv2Output
 - sc2o, gc2o, 194
- CMRv2Usage
 - setCMRv2Usage, getCMRv2Usage
 - sc2u, gc2u, 195
- CN0Mask
 - setCN0Mask, getCN0Mask
 - scm, gcm, 92
- CommandHelp
 - lstCommandHelp
 - help, 71
- COMSettings
 - setCOMSettings, getCOMSettings
 - scs, gcs, 148
- ConfigFile
 - lstConfigFile
 - lcf, 72
- CopyConfigFile
 - exeCopyConfigFile, getCopyConfigFile
 - eccf, gccf, 73
- CrossDomainWebAccess
 - setCrossDomainWebAccess, getCrossDomainWebAccess
 - scda, gcda, 149
- CurrentUser
 - lstCurrentUser
 - lcu, 84

D

- DataInOut
 - setDataInOut, getDataInOut
 - sdio, gdio, 150
- DefaultAccessLevel
 - setDefaultAccessLevel, getDefaultAccessLevel
 - sdal, gdal, 85
- DiffCorrMaxAge
 - setDiffCorrMaxAge, getDiffCorrMaxAge
 - sdca, gdca, 105
- DiffCorrUsage
 - setDiffCorrUsage, getDiffCorrUsage
 - sdcu, gdcu, 106
- DiskFullAction
 - setDiskFullAction, getDiskFullAction
 - sdfa, gdfa, 196
- DiskInfo
 - lstDiskInfo
 - ldi, 197

E

- EchoMessage
 - exeEchoMessage, getEchoMessage
 - eeem, gecm, 152
- ElevationMask

setElevationMask, getElevationMask

sem, gem, 107

EventParameters

setEventParameters, getEventParameters

sep, gep, 139

F

FileNaming

setFileNaming, getFileNaming

sfn, gfn, 198

FixReliability

setFixReliability, getFixReliability

sfr, gfr, 108

FrontendMode

setFrontendMode, getFrontendMode

sfm, gfm, 93

FTPPushRINEX

setFTPPushRINEX, getFTPPushRINEX

sfpr, gfpr, 200

FTPPushSBF

setFTPPushSBF, getFTPPushSBF

sfps, gfps, 201

FTPUpgrade

exeFTPUpgrade, getFTPUpgrade

efup, gfup, 74

G

GeodeticDatum

setGeodeticDatum, getGeodeticDatum

sgd, ggd, 109

GeoidUndulation

setGeoidUndulation, getGeoidUndulation

sgu, ggu, 111

GNSSAttitude

setGNSSAttitude, getGNSSAttitude

sga, gga, 112

GPIOFunctionality

setGPIOFunctionality, getGPIOFunctionality

sgpf, ggpf, 140

H

HealthMask

setHealthMask, getHealthMask

shm, ghm, 113

I

InternalFile

lstInternalFile

lif, 75

IonosphereModel

setIonosphereModel, getIonosphereModel

sim, gim, 114

IPPortSettings

setIPPortSettings, getIPPortSettings

sipp, gipp, 153

IPReceiveSettings

setIPReceiveSettings, getIPReceiveSettings

sirs, girs, 154

IPServerSettings

setIPServerSettings, getIPServerSettings

siss, giss, 155

IPSettings

setIPSettings, getIPSettings

sips, gips, 156

L

LBandBeams

lstLBandBeams

llbb, 207

setLBandBeams, getLBandBeams

slbb, glbb, 208

LBandSelectMode

setLBandSelectMode, getLBandSelectMode

slsm, glsm, 209

LBAS1PPPConfig

setLBAS1PPPConfig, getLBAS1PPPConfig

slpc, glpc, 210

LBAS1RefStations

lstLBAS1RefStations

llrs, 211

setLBAS1RefStations, getLBAS1RefStations

slrs, glrs, 212

LEDMode

setLEDMode, getLEDMode

slm, glm, 141

LogIn

LogIn

login, 86

LogOut

LogOut

logout, 87

M

MagneticVariance

setMagneticVariance, getMagneticVariance

smv, gmv, 115

ManageDisk

exeManageDisk, getManageDisk

emd, gmd, 202

MarkerParameters

setMarkerParameters, getMarkerParameters

smp, gmp, 145

MIBDescription

lstMIBDescription

lmd, 76

MultipathMitigation
 setMultipathMitigation, getMultipathMitigation
 smm, gmm, 94

N

NetworkRTKConfig
 setNetworkRTKConfig, getNetworkRTKConfig
 snrc, gnrc, 116

NMEAOnce
 exeNMEAOnce, getNMEAOnce
 enoc, gnoc, 157

NMEAOutput
 setNMEAOutput, getNMEAOutput
 sno, gno, 158

NMEAPrecision
 setNMEAPrecision, getNMEAPrecision
 snp, gnp, 160

NMEATalkerID
 setNMEATalkerID, getNMEATalkerID
 snti, gnti, 161

NMEAVersion
 setNMEAVersion, getNMEAVersion
 snv, gnv, 162

NotchFiltering
 setNotchFiltering, getNotchFiltering
 snf, gnf, 95

NtripSettings
 setNtripSettings, getNtripSettings
 snts, gnts, 163

NTRIPSourceTable
 lstNTRIPSourceTable
 lnst, 164

NWALevels
 setNWALevels, getNWALevels
 snl, gnl, 117

O

ObserverComment
 setObserverComment, getObserverComment
 soc, goc, 146

ObserverParameters
 setObserverParameters, getObserverParameters
 sop, gop, 147

P

PeriodicEcho
 setPeriodicEcho, getPeriodicEcho
 spe, gpe, 165

PowerMode
 exePowerMode, getPowerMode
 epwm, gpwm, 77

PPPAutoSeed

setPPPAutoSeed, getPPPAutoSeed
 spas, gpas, 213
 PPPDatumOffset
 setPPPDatumOffset, getPPPDatumOffset
 spdo, gpdo, 214
 PPPSetSeedGeod
 exePPPSetSeedGeod, getPPPSetSeedGeod
 epss, gpss, 118
 PPSPParameters
 setPPSPParameters, getPPSPParameters
 spps, gpps, 142
 PVTMode
 setPVTMode, getPVTMode
 spm, gpm, 120

R

RAIMLevels
 setRAIMLevels, getRAIMLevels
 srl, grl, 122
 ReceiverCapabilities
 getReceiverCapabilities
 grc, 78
 ReceiverDynamics
 setReceiverDynamics, getReceiverDynamics
 srd, grd, 123
 ReceiverInterface
 getReceiverInterface
 gri, 80
 RecordedFile
 lstRecordedFile
 lrf, 203
 RegisteredApplications
 exeRegisteredApplications, getRegisteredApplications
 era, gra, 81
 RemoveFile
 exeRemoveFile, getRemoveFile
 erf, grf, 204
 ResetNavFilter
 exeResetNavFilter, getResetNavFilter
 ernf, grnf, 124
 ResetReceiver
 exeResetReceiver, getResetReceiver
 erst, grst, 82
 RINEXLogging
 setRINEXLogging, getRINEXLogging
 srxl, grxl, 205
 RTCMv2Compatibility
 setRTCMv2Compatibility, getRTCMv2Compatibility
 sr2c, gr2c, 173
 RTCMv2Formatting
 setRTCMv2Formatting, getRTCMv2Formatting
 sr2f, gr2f, 174

RTCMv2Interval
 setRTCMv2Interval, getRTCMv2Interval
 sr2i, gr2i, 175

RTCMv2IntervalObs
 setRTCMv2IntervalObs, getRTCMv2IntervalObs
 sr2b, gr2b, 176

RTCMv2Message16
 setRTCMv2Message16, getRTCMv2Message16
 sr2m, gr2m, 177

RTCMv2Output
 setRTCMv2Output, getRTCMv2Output
 sr2o, gr2o, 178

RTCMv2Usage
 setRTCMv2Usage, getRTCMv2Usage
 sr2u, gr2u, 179

RTCMv3CRSTransfo
 setRTCMv3CRSTransfo, getRTCMv3CRSTransfo
 sr3t, gr3t, 180

RTCMv3Delay
 setRTCMv3Delay, getRTCMv3Delay
 sr3d, gr3d, 181

RTCMv3Formatting
 setRTCMv3Formatting, getRTCMv3Formatting
 sr3f, gr3f, 182

RTCMv3Interval
 setRTCMv3Interval, getRTCMv3Interval
 sr3i, gr3i, 183

RTCMv3Message1029
 setRTCMv3Message1029, getRTCMv3Message1029
 sr3m, gr3m, 184

RTCMv3Output
 setRTCMv3Output, getRTCMv3Output
 sr3o, gr3o, 185

RTCMv3Usage
 setRTCMv3Usage, getRTCMv3Usage
 sr3u, gr3u, 188

S

SatelliteTracking
 setSatelliteTracking, getSatelliteTracking
 sst, gst, 96

SatelliteUsage
 setSatelliteUsage, getSatelliteUsage
 ssu, gsu, 125

SBASCORRECTIONS
 setSBASCORRECTIONS, getSBASCORRECTIONS
 ssbc, gsbc, 127

SBFGROUPS
 setSBFGROUPS, getSBFGROUPS
 ssgp, gsgp, 167

SBFOnce
 exeSBFOnce, getSBFOnce

esoc, gsoc, 168

SBFOutput

setSBFOutput, getSBFOutput

sso, gso, 170

SignalTracking

setSignalTracking, getSignalTracking

snt, gnt, 98

SignalUsage

setSignalUsage, getSignalUsage

snu, gnu, 128

SmoothingInterval

setSmoothingInterval, getSmoothingInterval

ssi, gsi, 99

StaticPosCartesian

setStaticPosCartesian, getStaticPosCartesian

sspc, gspc, 129

StaticPosGeodetic

setStaticPosGeodetic, getStaticPosGeodetic

sspg, gspg, 130

T

TimingSystem

setTimingSystem, getTimingSystem

sts, gts, 131

TrackingLoopParameters

setTrackingLoopParameters, getTrackingLoopParameters

stlp, gtlp, 100

TroposphereModel

setTroposphereModel, getTroposphereModel

stm, gtm, 132

TroposphereParameters

setTroposphereParameters, getTroposphereParameters

stp, gtp, 134

U

UserAccessLevel

setUserAccessLevel, getUserAccessLevel

sual, gual, 88

UserDatum

setUserDatum, getUserDatum

sud, gud, 135

UserDatumVel

setUserDatumVel, getUserDatumVel

sudv, gudv, 136

UserEllipsoid

setUserEllipsoid, getUserEllipsoid

sue, gue, 137

W

WakeUpInterval

setWakeUpInterval, getWakeUpInterval

swui, gwui, 143

chapterIndex of SBF Blocks

ASCIIn, 458
AttCovEuler, 386
AttEuler, 383

BaseLine, 461
BaseStation, 409
BaseVectorCart, 373
BaseVectorGeod, 377
BBSamples, 456

ChannelStatus, 423
CMPNav, 293
CMPRaw, 258
Commands, 453
Comment, 454

DiffCorrIn, 406
DiskStatus, 449
DOP, 348

EndOfAtt, 389
EndOfMeas, 241
EndOfPVT, 382
ExtEvent, 395
ExtEventPVTCartesian, 398
ExtEventPVTGeodetic, 402

GALAlm, 283
GALGstGps, 289
GALIon, 285
GALNav, 280
GALRawFNAV, 250
GALRawINAV, 252
GALSARLM, 291
GALUtc, 287
GEOAlm, 312
GEOClockEphCovMatrix, 322
GEOCorrections, 371
GEODegrFactors, 308
GEOFastCorr, 301
GEOFastCorrDegr, 305
GEOIGPMask, 314
GEOIntegrity, 304
GEOIonoDelay, 318
GEOLongTermCorr, 316
GEOMT00, 298
GEONav, 306
GEONetworkTime, 310
GEOPRNMask, 299
GEORawL1, 254
GEORawL5, 256
GEOServiceLevel, 320

GLOAlm, 276
GLONav, 274
GLORawCA, 248
GLOTime, 278
GPSAlm, 268
GPSIon, 270
GPSNav, 266
GPSRawCA, 242
GPSRawL2C, 244
GPSRawL5, 246
GPSUtc, 272

InputLink, 434
IPStatus, 444
IQCorr, 237
ISMR, 239

LBandBeams, 421
LBandTrackerStatus, 415
LBAS1DecoderStatus, 417
LBAS1Messages, 419

MeasEpoch, 228
MeasExtra, 235

NTRIPClientStatus, 442

OutputLink, 438

PosCart, 351
PosCovCartesian, 336
PosCovGeodetic, 339
PosLocal, 355
PosProjected, 358
PVTCartesian, 324
PVTGeodetic, 330
PVTResiduals, 364
PVTSatCartesian, 362
PVTSupport, 381

QualityInd, 446
QZSNav, 295
QZSRawL1CA, 260
QZSRawL2C, 262
QZSRawL5, 264

RAIMStatistics, 369
ReceiverSetup, 451
ReceiverStatus, 427
ReceiverTime, 390
RTCMDatum, 412

SatVisibility, 432

VelCovCartesian, 342

VelCovGeodetic, 345

xPPSOffset, 392

Appendix A

Attitude Angles

Appendix A.1 Vehicle Reference Frame

The vehicle reference frame is attached to the vehicle. It has its X axis pointing along the longitudinal vehicle axis, the Y axis pointing towards the vehicle starboard (right) side and the Z axis pointing down, as illustrated in figure A.1-1.

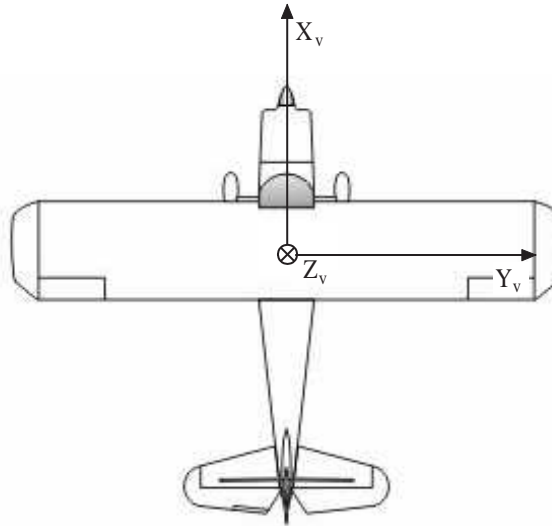


Figure A.1-1: Vehicle frame.

The attitude of the vehicle is defined as the angles between the vehicle frame and the local-level reference frame (defined by the East, North and Up directions). Septentrio receivers express the vehicle attitude in Euler angles using the heading-pitch-roll rotation sequence.

Appendix A.2 Euler Angles

Euler angles are defined as successive rotations of the vehicle frame (X, Y, Z axes) relative to the local-level East-North-Up reference frame. The rotation sequence is shown in figure A.2-1. The heading (ψ) of the vehicle is defined as the right-handed rotation of the vehicle about the Z axis ($0^\circ \leq \psi \leq 360^\circ$). The pitch (θ) of the vehicle is defined as the right-handed rotation about the vehicle Y axis ($-90^\circ \leq \theta \leq 90^\circ$). The roll (ϕ) of the vehicle is defined as the right-handed rotation about the vehicle X axis ($-180^\circ \leq \phi \leq 180^\circ$).

Starting from the situation where X points to the North, Y to the East and Z down, the following successive rotations define the attitude of the vehicle. Note that the order of the rotations is important.

1. Rotate through angle ψ about Z axis;
2. Rotate through angle θ about new Y axis;
3. Rotate through angle ϕ about new X axis;

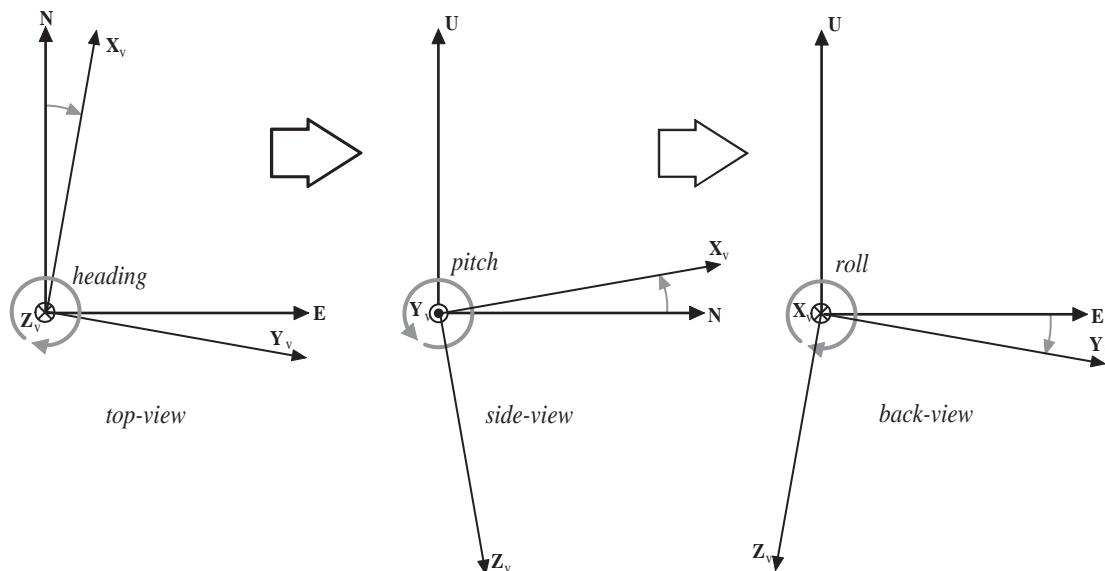


Figure A.2-1: Euler angle sequence.

Appendix B

NMEA Overview

The following tables provide a list of supported NMEA messages. For a full description of these messages, please refer to the respective standard.

NMEA Sentence	Short Description	Comment
ALM	GPS Almanac Data	
AVR	Trimble Navigation proprietary \$PTNL, AVR sentence	
DTM	Datum Reference	
GBS	GNSS Satellite Fault Detection	
GFA	GNSS Fix Accuracy and Integrity	Use setNWAlevels command to set the accuracy level
GGA	GPS Fix Data	GPS Quality Indicator field is set to 5 in PPP mode
GGK	Trimble Navigation proprietary \$PTNL, GGK sentence	
GGQ	Leica Real-Time Position with CQ	
GLL	Geographic Position - Latitude/Longitude	
GMP	GNSS Map Projection Fix Data	
GNS	GNSS Fix Data	
GRS	GNSS Range Residuals	
GSA	GNSS DOP and Active Satellites	
GST	GNSS Pseudorange Error Statistics	
GSV	GNSS Satellites in View	
HDT	Heading, True	
HRP	Heading, Roll, Pitch	Septentrio proprietary, see section B.1
LLK	Leica Local Position and GDOP	
LLQ	Leica Local Position and Quality	
PUMRD	Septentrio proprietary	Septentrio proprietary, not documented here
RBD	Rover-Base Direction	Septentrio proprietary, see section B.1
RBP	Rover-Base Position	Septentrio proprietary, see section B.1
RBV	Rover-Base Velocity	Septentrio proprietary, see section B.1
RMC	Recommended Minimum Specific GNSS Data	
ROT	Rate of Turn	
TXT	Text Transmission	Text from a base station. The text identifier is set to 1, and the text message is in the form " nnnn :<base txt>", where nnnn is the base station ID.
VTG	Course Over Ground and Ground Speed	
ZDA	Time and Date	

Appendix B.1 Proprietary NMEA Sentences

PSSN,HRP	Septentrio Proprietary - Heading, Roll, Pitch
Field	Description
\$PSSN,HRP,	Start of sentence
hhmmss.ss,	UTC of HRP (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Heading, degrees True
x.x,	Roll, degrees
x.x,	Pitch, degrees
x.x,	Heading standard deviation, degrees
x.x,	Roll standard deviation, degrees
x.x,	Pitch standard deviation, degrees
xx,	Number of satellites used for attitude computation
x,	Mode indicator: 0: No attitude available 1: Heading, Pitch with float ambiguities 2: Heading, Pitch with fixed ambiguities 3: Heading, Pitch, Roll with float ambiguities 4: Heading, Pitch, Roll with fixed ambiguities 5: Heading, Pitch from velocity (dead-reckoning)
x.x,a	Magnetic variation, degrees (E=East, W=West, see also the setMagneticVariance command)
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBP Septentrio Proprietary - Rover-Base Position	
Field	Description
\$PSSN,RBP,	Start of sentence
hhmmss.ss,	UTC of RBP (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	North (True) baseline component (positive when base is north of rover), meters
x.x,	East baseline component (positive when base is east of rover), meters
x.x,	Up baseline component (positive when base is higher than rover), meters
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c-c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBD Septentrio Proprietary - Rover-Base Direction	
Field	Description
\$PSSN,RBD,	Start of sentence
hhmmss.ss,	UTC of RBD (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Azimuth of the base as seen from rover (0 to 360 increasing towards east), degrees True
x.x,	Elevation of the base as seen from rover (-90 to 90), degrees
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c-c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBV Septentrio Proprietary - Rover-Base Velocity	
Field	Description
\$PSSN,RBV,	Start of sentence
hhmmss.ss,	UTC of RBV (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Rate of change of baseline vector (rover to base), north component, m/s
x.x,	Rate of change of baseline vector (rover to base), east component, m/s
x.x,	Rate of change of baseline vector (rover to base), up component, m/s
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c-c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

Appendix C

Error Messages

The following table lists the possible ASCII error messages and their meaning.

Error Message	Description
Invalid command!	Syntax error or unsupported command.
Argument 'xxx' can't be omitted!	Omission of a mandatory argument.
At least one non tabular argument needed!	Omission of a mandatory argument.
Argument 'xxx' is invalid!	Value out of range, or too many decimal digits.
Argument 'xxx' could not be handled!	Argument impossible to parse or invalid combination of values.
ASCII commands between prompts were discarded!	Argument impossible to parse or invalid combination of values.