Altus NR2 Reference Guide

Applicable to version 1.1.0 of the Control Firmware

Altus NR2 Reference Guide

September 18, 2015

Applicable to version 1.1.0 of the Control Firmware

Septentrio
Greenhill Campus, Interleuvenlaan 15G
B-3001 Leuven, Belgium

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

List of Contents

Scope

This document contains reference information about the receiver firmware.

Chapter 1 provides a set of step-by-step "how-to's" to help you find your way around the receiver's commands and logs.

Chapter 2 provides some background on the main algorithms running in the receiver and on the way to configure them.

Chapter 3 contains the complete description of the user command interface.

Chapter 4 contains the complete description of the SBF format.

Typographical Conventions

`abc` User command name. Clicking a command name redirects to the full command description.

abc Command argument name.

`abc` Command replies.

SBF block name or SBF field name. Clicking a SBF block name redirects to the full SBF block description.

List of Acronyms

Abbreviation	Description
AGC	Automatic Gain Control
ARP	Antenna Reference Point
ASCII	American Standard Code for Information Interchange
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
DOP	Dilution Of Precision
DVS	Data Validity Status
ECEF	Earth-Centered Earth-Fixed
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
GUI	Graphical User Interface
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
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
ITRS	International Terrestrial Reference System
LBand	L-Band Receiver
L1	L1 carrier
L2	L2 carrier
L2C	L2C code
LSB	Least Significant Bit
MDB	Minimum Detectable Bias
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
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
UHF	Ultra High Frequency
URA	User Range Accuracy
USB	Universal Serial Bus
UTC	Coordinated Universal Time
VDOP	Vertical DOP
VERL	Vertical External Reliability Level
VPL	Vertical Protection Level
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

How To...

This chapter contains step-by-step instructions to help you with typical tasks. It will help you to familiarize yourself with the receiver commands without going into too much detail.

For a comprehensive description of the command set, refer to chapter 3. You can also click on any command or SBF block name in this manual to be redirected to the full description of that command or SBF block.

You can enter user commands in many different ways:

- Commands can be accessed graphically through menus in RxControl and in the web interface (see section 1.1.5).
- Using the Data Link program provided in the RxTools suite (or any suitable terminal emulation program), you can enter commands manually through one of the receiver input ports (see section 1.1). 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`**.
- You can type commands or mnemonics in the console window of RxControl (menu *Tools* > *Expert Console*) or of the web interface (menu *Admin* > *Expert Console*).

⚠ Depending on the capabilities of your particular receiver (see section 1.15), some of the features described here may not be supported.

1.1 Connect to the Receiver

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

1.1.2 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 diver can be installed through the RxTools software suite. On Linux, the standard Linux CDC-ACM driver is suitable. 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.

1.1.3 Via Bluetooth

By default, the Bluetooth device name of the receiver is "`Altus NR2-`*`xxxxxxx`*" where *`xxxxxxx`* is the serial number. The default Bluetooth pairing code is `1234`.

Bluetooth parameters can be modified with the **setBTParameters** command.

1.1.4 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**).

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.

TCP/IP connections can be made over the following interfaces.

1.1.4.1 WiFi Interface

1.1.4.1.1 Receiver is WiFi Access Point

By default, the receiver is configured in WiFi access point mode, with the SSID set to "`Altus NR2-`*`xxxxxxx`*" where *`xxxxxxx`* is the serial number. Encryption is disabled by default.

When you are connected to the receiver WiFi access point, the receiver can be reached at the fixed IP address `192.168.20.1`, or with the hostname `altus nr2` (depending on your network configuration, you may need to use `altus nr2.local` instead).

WiFi can be turned on and off with the **`setWiFiMode`** command, and the access point parameters (SSID, encryption, channel number, ...) can be adjusted with the **`setWiFiAccessPoint`** command.

1.1.4.1.2 Receiver is WiFi Client

It is possible to configure the receiver in WiFi client mode with the **`setWiFiMode`** command. In client mode, the receiver will attempt to connect to a reachable access point. The access point password must be entered with the command **`exeAddWiFiAccessPoint`** (this must be done only once). The list of access points can be obtained with the **`lstWiFiAccessPoints,all`** command.

In client mode, the receiver gets its IP address dynamically, and its hostname is `altus nr2-`*`xxxxxxx`*.

1.1.4.2 Ethernet-over-USB Interface

When an USB cable is connected, the receiver also supports Ethernet-over-USB. The IP address allocated to the Ethernet-over-USB interface is `192.168.3.1`. That address cannot be changed, so that this feature is only to be used when a single receiver is connected to your computer.

1.1.5 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 1.1.4.

For example, if your receiver's hostname is `altus nr2-1234567`, simply use the following URL in your preferred web browser:

```
http://altus nr2-1234567
```

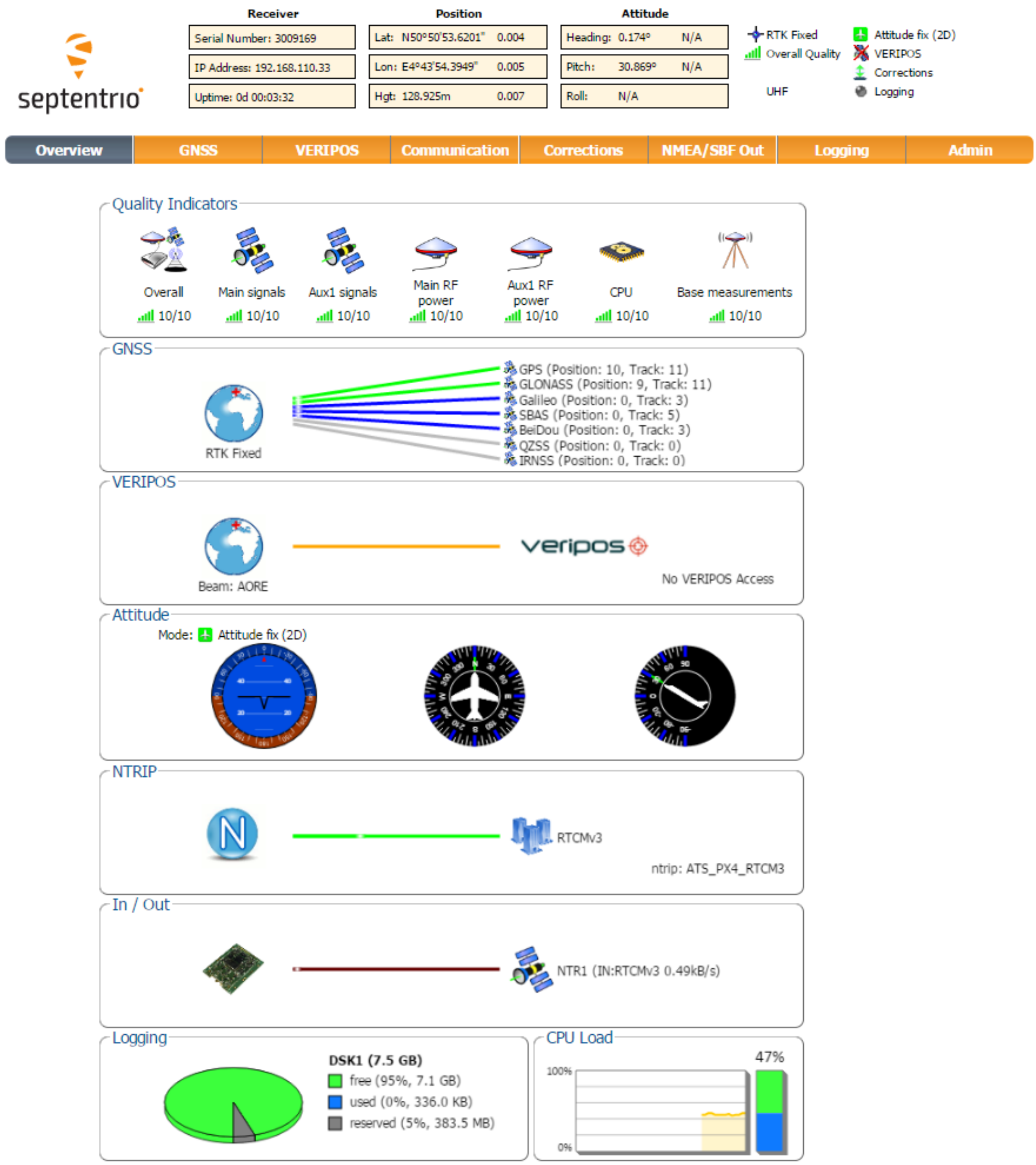

Figure 1-1: Web interface main window.

Most user commands described in section 3.2 can be accessed graphically from the web interface (*Admin > Expert Control Panel*). You can also go to *Admin > Expert Console* to manually type ASCII commands and view replies.

1.1.6 Connection Descriptors

Receiver connections are identified by their connection descriptor (CD). The different connection descriptors are shown in the table below. The three rigthmost columns indicate the direction (input or output or both), and whether the connection can accept user command input.

CD	Description	In	Out	Cmd
`COMx`	one of the serial ports	•	•	•
`USBx`	one of the virtual serial ports, built on top of the USB interface	•	•	•
`DSK1`	the internal disk. See section 1.12	•		
`IP1x`	one of the TCP/IP connections	•	•	•
`NTRx`	one of the NTRIP connections. Output in NTRIP server mode (section 1.7), input in NTRIP client mode (section 1.8)	•	•	
`IPSx`	one of the IP server connections. See section 1.9		•	
`IPRx`	one of the IP receive connections. See section 1.10	•		
`BT01`	the Bluetooth connection	•	•	•

For instance, to output the ASCII textual status screen to COM1, use:
`setDataInOut,COM1,,ASCIIDisplay <CR>`

1.2 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	`$R`
ASCII transmissions (e.g. periodic output of the status screen), terminated by a prompt. Two sub-types are defined: • `$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)	`$@`

1.2.1 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 section 4 for a description of the SBF format.

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 can be found in appendix B

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 NMEA

The receiver can generate a set of approved NMEA sentences, which conform to the NMEA Standard (version 2.30(1) and version 4.10(2) are supported). The benefit of the NMEA format is that it is standardized. Many electronic devices and software packages support NMEA. The drawback of NMEA is a relatively low level of detail.

NMEA output is configured with the **`setNMEAOutput`** command, and the NMEA version (2.30 or 4.10) is selected with the **`setNMEAVersion`** command.

The list of supported NMEA sentences can be found in appendix C.

(1) NMEA 0183, Standard for Interfacing Marine Electronic Devices, Version 2.30, National Marine Electronics Association, 1998
(2) NMEA 0183, Standard for Interfacing Marine Electronic Devices, Version 4.10, National Marine Electronics Association, 2012

1.2.3 RTCM and CMR

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 1.5.

The list of supported RTCM and CMR messages can be found in appendix D.

1.3 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* > *RxControl Logging* menu to access the logging configuration window. Logging on the receiver's internal disk is described in section 1.12.

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` SBF block at 10 Hz and the `PVTCartesian` SBF block at 1 Hz. In this example, we will assume that these 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>

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. Defining these SBF streams involves the **setSBFOutput** command:
   setSBFOutput,Stream1,USB2,MeasEpoch,msec100 <CR>
   setSBFOutput,Stream2,USB2,PVTCartesian,sec1 <CR>

   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,msec100 <CR>
   setSBFOutput,Stream4,COM1,PVTCartesian,sec1 <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>

1.4 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 section 3.2

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>

To revert to the default setting where the receiver starts in the default configuration, you should use:
exeCopyConfigFile,RxDefault,Boot <CR>

1.5 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 supported RTCM and CMR messages can be found in appendix D.

1.5.1 Static Base Station Mode

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

1. 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 2.1.4). For example, assuming the WGS84 position of the ARP is 50.5^{o}N, 4^{o}E and its altitude above the WGS84 ellipsoid is 100m, use:
   `setStaticPosGeodetic,Geodetic1,50.5,4,100 <CR>`
   `setPVTMode,Static, ,Geodetic1 <CR>`

   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>`

2. 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 2.1.4). For instance:
   `setAntennaOffset,Main, , , ,"AT2775-54SW","5684" <CR>`

3. Use the commands **`setRTCMv2Interval`**, **`setRTCMv2IntervalObs`**, **`setRTCM-v3Interval`** or **`setCMRv2Interval`** to specify the message interval (default is one second for most messages). For instance, to change the interval at which RTCM 3.x message type 1033 is generated to 10 seconds, type:
   `setRTCMv3Interval,RTCM1033,10 <CR>`

4. Use the commands **`setRTCMv2Formatting`**, **`setRTCMv3Formatting`** or **`set-CMRv2Formatting`** 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:
   `setRTCMv3Formatting,496 <CR>`

5. By default, the receiver is configured to output all RTCM and CMR messages necessary for DGPS and RTK operation. If necessary, the set of output messages can be specified with the commands **`setRTCMv2Output`**, **`setRTCMv3Output`** or **`setCMRv2Output`**. For instance, to output RTCM3.x messages 1006, 1033 and 1074 on COM2, use:
   `setRTCMv3Output,COM2,RTCM1006+RTCM1033+RTCM1074 <CR>`

   If you are using the RTCM3.x MSM messages (see appendix D), you can use the **`setRTCMv3Formatting`** command to configure the signal types that need to be encoded in MSM.

6. The RTCM stream can be output through any output connection listed in section 1.1.6. For instance, to enable RTCM 3.x output through COM2, use:
setDataInOut,COM2, ,RTCMv3 <CR>

7. When sending differential corrections over a serial port, do not forget to specify the baud rate. For instance if the differential correction stream needs to be output on COM2 at 9600 baud, use:
setCOMSettings,COM2,baud9600 <CR>

To stop transmitting RTCM messages, enter the following command:
setDataInOut,COM2, ,none <CR>

Note that, even in static mode, the receiver computes a PVT solution to estimate the clock bias.

1.6 Configure the Receiver in DGPS/RTK-Rover Mode

The receiver computes a DGPS and/or an RTK solution when it receives the relevant differential correction messages on one of its connections. The list of supported differential correction messages can be found in appendix D.

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

1. Make sure that at least one of the receiver connections is receiving differential corrections. Any input connection listed in section 1.1.6 is suitable. When using a serial connection, 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>`

2. By default, the receiver assumes that the base station is static. If it is moving, enter the following command:
   `setDiffCorrUsage,,,,,on <CR>`
   In DGPS-rover mode, the base station must be static. Moving base stations are only supported in RTK-rover mode.

3. The receiver automatically detects the format of the differential corrections (RTCM or CMR) and switches between standalone, DGPS or RTK modes according to the type of corrections it receives, provided these modes are enabled with the **`setPVTMode`** command (all modes are enabled by default).

Refer to sections 2.1.2 and 2.1.3 for further details on the DGPS and RTK positioning mode.

1.7 Configure the Receiver in NTRIP Server Mode

In the example below, we show how to configure the receiver to send RTCM 3.x 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 1.1.6) in server mode for sending data to 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>`

2. By default, for RTCM 3.x, the receiver is configured to send message types 1004, 1006, 1012 ans 1033 at an interval of one second. This can be changed by using the **`setRTCMv3Output`** and **`setRTCMv3Interval`** commands. For instance, to change the interval of RTCM1033 to 10 seconds, use:
   `setRTCMv3Interval,RTCM1033,10 <CR>`

3. Enable the output of RTCM 3.x corrections on the `NTR1` connection:
   `setDataInOut,NTR1, ,RTCMv3 <CR>`

4. Closing the NTRIP connection is done with the following command:
   `setNTRIPSettings,NTR1,off <CR>`

See also section 1.5 for more information on configuring the receiver as a base station.

1.8 Configure the Receiver in NTRIP Client Mode

In this section, we show how to configure the receiver to receive and use RTK corrections from an NtripCaster. In the example below, the NtripCaster and Mount Point details are as follows:

- 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 1.1.6) for communication with the NtripCaster in client mode. Here, we assume that the first NTRIP connection (`NTR1`) is free and can be used for that purpose:
   `setNTRIPSettings,NTR1,Client,ntrip.example.com,2101,USER,PASSWD,LEUV1 <CR>`

2. The receiver will automatically receive and decode the RTK corrections from the NtripCaster and switch to RTK positioning mode, unless RTK is disabled with the **`setPVTMode`** command.

3. Closing the NTRIP connection is done with the following command:
   `setNTRIPSettings,NTR1,off <CR>`

The status of the NTRIP client connection is reported in the `NTRIPClientStatus` SBF block.

1.9 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 1.1.6) to listen to port 28785. Here, we assume that the first IP server connection (`IPS1`) is free:
   `setIPServerSettings,IPS1,28785,TCP <CR>`

2. Output the GGA NMEA message to the `IPS1` connection, at a 1-Hz rate:
   `setNMEAOutput,Stream1,IPS1,GGA,sec1 <CR>`

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>`

4. You will need to reset the receiver for the new IP server configuration to take effect:
   `exeResetReceiver,Soft,none <CR>`

A way to check the IP server functionality is to enter the URL **`http://`*`altus nr2-xxxxxxx`*`:28785`** in your preferred web browser (replace *`altus nr2-xxxxxxx`* 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.

The example above showed how to set up a TCP server. It is also possible to configure the receiver in UDP server mode. For example, to broadcast the GGA message to any UDP client listening to its port 28785, the command in step 1. above must be replaced by:
`setIPServerSettings,IPS1,28785,UDP,255.255.255.255 <CR>`

Conversely, the receiver can be configured to automatically receive data from an IP server. This is explained in section 1.10.

1.10 Configure an IP Receive Port

The receiver can be configured to automatically receive data (typically differential corrections) from an IP server. In this example, we show how to connect to an IP server having the hostname `MyServer` and using port 28786.

1. Configure one of the IP receive connections (see section 1.1.6) to listen to port 28786 of `MyServer`. Here, we assume that the first IP receive connection (`IPR1`) is free:
   `setIPReceiveSettings,IPR1,28786,TCP,MyServer <CR>`

2. If the data stream from the IP server contains differential corrections in CMR or RTCM format, the receiver will automatically decode them and use them in the PVT processing.

3. To close the connection, enter the following command:
   `setIPReceiveSettings,IPR1,0 <CR>`

The example showed how to set up a TCP connection with the server. The receiver can also listen to incoming UDP messages. In that case, the server address or hostname must not be specified. For example, to listen to UDP messages on port 28786, use the command:
`setIPReceiveSettings,IPR1,28786,UDP <CR>`

1.11 Use the Cellular Modem

The cellular modem allows connecting to an NTRIP caster (see section 1.8). However, remote control of the receiver over the cellular modem is not supported.

Enabling the cellular modem involves the following steps:

1. Make sure a SIM card with data subscription is installed in the receiver.

2. Use the **setCellularParameters** command to turn on the cellular modem and allow it to connect to internet. For example, if your access point name (APN) is `myapn.com`, enter the following command:
   setCellularParameters,on,on,myapn.com <CR>

3. The first time you use the cellular modem, or after changing the SIM card, you will need to enter the PIN with the **setCellularPIN** command:
   setCellularPIN,1234 <CR>

The commands **exeChangeCellularPIN** is available to change the PIN code, and the command **exeUnblockCellular** can be used to enter the PUK code if necessary.

The status of the cellular modem connection is reported in the `CellularStatus` SBF block.

1.11.1 Mobile Hotspot Configuration

When configured in WiFi access point mode (see section 1.1.4.1.1), the receiver can serve as a mobile hotspot and provide access to the internet through the cellular modem. For example, to enable a WiFi mobile hotspot with password "`MyPassword`", enter the following command:
setWiFiAccessPoint, ,WPA2,MyPassword, ,on <CR>

⚠ When enabling the mobile hotspot, make sure to secure your WiFi access point by providing an encryption type and a password, as shown in the above example.

1.12 Log SBF or NMEA on the Internal Disk

The connection descriptor (see section 1.1.6) associated to the internal disk is "`DSK1`". Enabling SBF or NMEA logging on the internal disk 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>`

   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 1.3). 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>`

3. Start the logging by enabling SBF and NMEA output to the `DSK1` connection (it is enabled by default):
   `setDataInOut,DSK1, ,+SBF+NMEA <CR>`

4. Once the logging session is finished, stop the logging by invoking:
   `setDataInOut,DSK1, ,-SBF-NMEA <CR>`

Refer to section 1.13 to learn how to download the logged files.

1.13 Download Log Files from the Receiver

There are different ways to download or delete files from the internal disk:

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

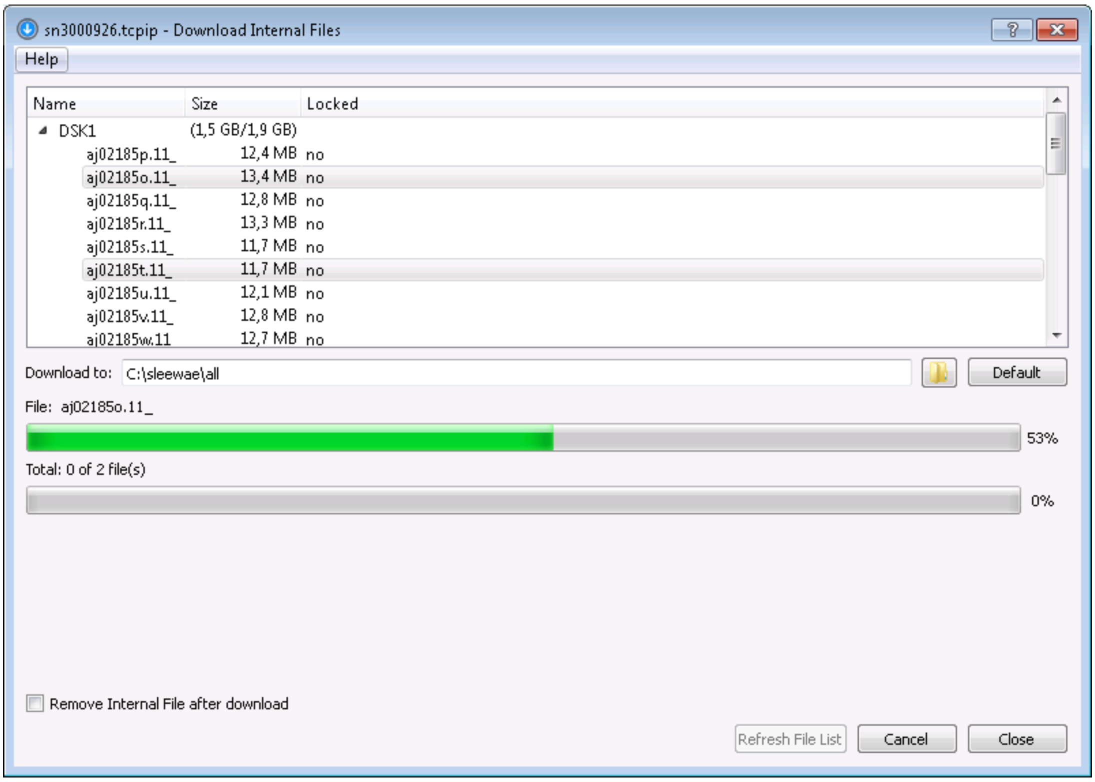

Figure 1-2: Download Internal Files from RxControl.

2. Through FTP. The hostname or fixed IP address is defined as explained in section 1.1.4. For example, if your receiver's hostname is `altus nr2-1234567`, you could type the following URL in your preferred web browser to open a FTP session as anonymous user:
   `ftp://altus nr2-1234567`

   The log files are found under the directory `ssn/SSRC9`.

3. Using the web interface. The web interface allows you to view the contents of the internal disk and to delete or download files (select the *Logging* tab).

4. By entering commands manually: the command **`lstDiskInfo`** prints the disk contents and free space and the command **`exeRemoveFile`** can be used to remove a file.

1.14 Upgrade the Receiver

Upgrading the receiver is the process of installing a new GNSS firmware, a new permission file (see section 1.16) or a new antenna calibration file (see section 2.1.4).

Upgrading the GNSS firmware can clear the receiver configuration stored in non-volatile memory (see section 1.4). 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 >Upgrade*).
4. 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>`

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.

1.15 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 1.16.

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

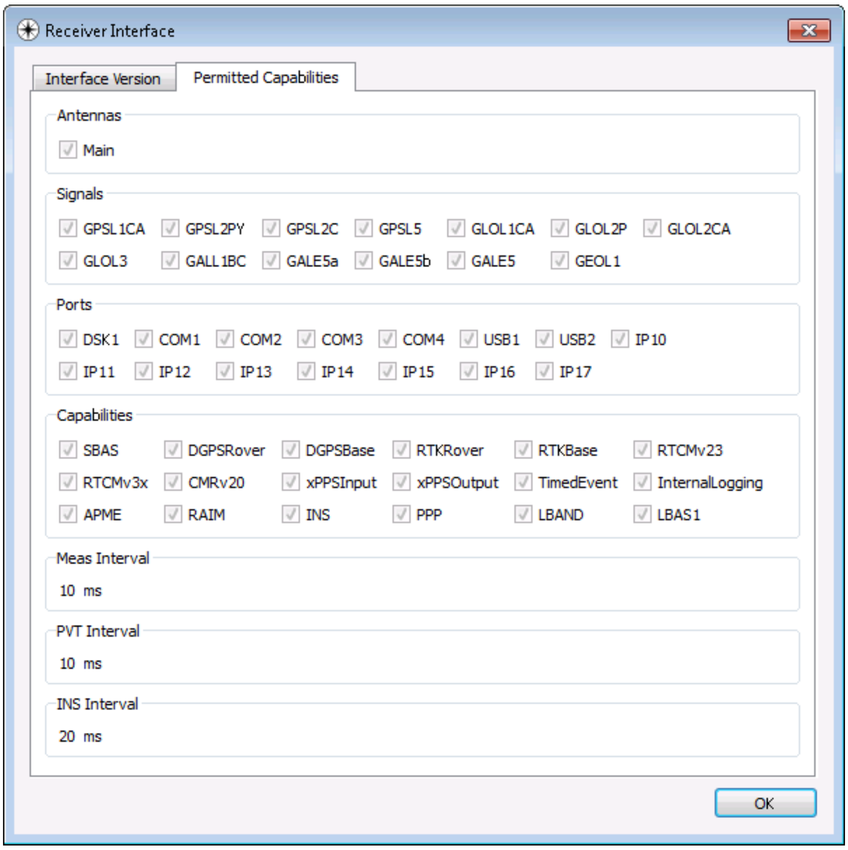

Figure 1-3: Example of receiver capabilities.

1.16 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 **`lstInternalFile, Permissions`**, or with RxControl by clicking *Help >Receiver Permissions*.

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 1.15.

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 1.14).

Chapter 2

Operation Details

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

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

The receiver computes the position, velocity and time (PVT) 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 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.

2.1.1 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).

2.1.2 DGPS Positioning (Single and Multi-Base)

DGPS (Differential GPS) is a pseudorange-based positioning technique where GNSS system errors are reduced by the use of range corrections. To work in DGPS rover mode, the receiver needs to receive differential corrections in the RTCM or CMR format.

2.1.3 RTK Positioning

Real-Time Kinematic (RTK) is a carrier phase positioning method where the carrier phase ambiguities are estimated in a kinematic mode.

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.

2.1.3.1 Integer Ambiguities (RTK-fixed)

The key to high-accuracy carrier phase positioning is the fixing of the carrier phase integer ambiguities. 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[(1)] developed at Delft University, department of Geodesy.

(1) 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.

2.1.3.2 Floating Ambiguities (RTK-float)

When data availability is low (e.g. 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.

2.1.4 Antenna Effects

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

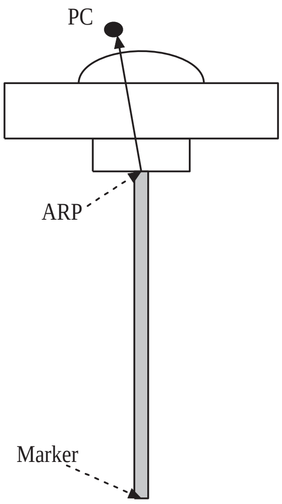

Figure 2-1: Antenna mount.

The GNSS measurements (pseudoranges and carrier phases observables) refer to a theoretical point in space called the phase center (noted PC in figure 2-1). 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. PC to ARP calibration tables are available on Internet for a large number of geodetic-grade antennas. For example, the National Geodetic Survey (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 1.14 (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 built-in 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 2-1). 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.

2.1.5 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.

Chapter 3

Command Line Reference

3.1 Command Line Interface 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 (see section1.1.6). 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.

3.1.1 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.

3.1.2 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.

- 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.

3.1.3 Command Syntax Tables

All ASCII commands are listed in section 3.2. 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 . . . 0.00 . . . 20.00 m	1 . . . 50 sec	Unknown	on off	none + G01 ... G32 + S120 ... S138 + SBAS + GPS all

GUI: 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 link printed in blue under the syntax table shows under which GUI menu the command can be found.

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

  In password arguments (arguments named *Password* or *Key*), special characters are allowed using the corresponding escape sequence:
  - Type `%%DQ` to obtain `"`
  - Type `%%SQ` to obtain `'`
  - Type `%%DL` to obtain `$`
  - Type `%%AM` to obtain `&`
  - Type `%%CM` to obtain `,`

  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`

3.2 Command Definitions

3.2.1 Receiver Administration

lcf	lstConfigFile	*File*								
		Current Boot RxDefault 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).
`RxDefault`	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 Boot User1 User2 RxDefault	Current Boot User1 User2							

RxControl: File > Copy Configuration

Use this command to manage the configuration files. See the **lstConfigFile** 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>
```

fwd	forward	*Destination*	*Command (255)*							
		Gnss_receiver Cell								

Use this command to forward a command to any device embedded in your product for which command forwarding is enabled. The target device must be specified with the *Destination* argument, and the command to be forwarded must be specified in the *Command* argument. The following target devices are supported:

Destination	Description
`Gnss_receiver`	forwards the command to the GNSS receiver incorporated in your product.
`Cell`	forwards the command to the cellular modem.

The commands must be compliant with the command interface of the target device. An error message is returned if the target device is unable to answer the command within a reasonable time.

This command is intended for expert-level users as it may interfer with the normal receiver operation.

Note that the **forward** command is not stored in the boot configuration, and is therefore not persistent across power-cycles.

Examples

```
COM1>  fwd,Gnss_receiver,help,Overview<CR>
$R; fwd,Gnss_receiver,Overview
---->
$-- BLOCK 0 / 0
$R; help, Overview
---->
$-- BLOCK 1 / 0
MENU: communication
  GROUP: ioSelection
    sdio, setDataInOut
    gdio, getDataInOut
...
COM1>

COM1>  fwd,Gnss_receiver,sdio,IP10,RTCMv2,NMEA<CR>
$R; fwd,Gnss_receiver,sdio,IP10,RTCMv2,NMEA
---->
$-- BLOCK 0 / 0
$R: sdio,IP10,RTCMv2,NMEA
  DataInOut, IP10, RTCMv2, NMEA
COM1>

COM1>  fwd,Cell,AT+CSQ<CR>
$R; fwd,Cell,AT+CSQ
---->
$-- BLOCK 0 / 0
```

```
AT+CSQ
+CSQ: 23,99
OK
COM1>
```

lif	lstInternalFile	*File*								
		Permissions Identification Debug Error								

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.

Example

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

grc	getReceiverCapabilities									

RxControl: 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.
`GLOL1CA`	GLONASS L1 C/A signal.
`GLOL2CA`	GLONASS L2 C/A signal.
`GALL1BC`	Galileo L1 BC signal.
`GEOL1`	SBAS L1 C/A signal.
`QZSL1CA`	QZSS L1 C/A signal.
`QZSL2C`	QZSS L2 C signal.

ComPorts	Description
`COM1`	Serial port 1.
`USB1`	Virtual serial port 1.
`USB2`	Virtual serial port 2.
`IP10`	TCP/IP port 1.
`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.

ComPorts (Continued)	Description
NTR1	NTRIP port 1.
IPS1	IP Server port 1.
IPS2	IP Server port 2.
IPS3	IP Server port 3.
BT01	Bluetooth serial port 1.
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.
InternalLogging	Internal logging.

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

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 **lstInternalFile, 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 + USB1 + USB2 + IP10 ... IP17 + BT01 all	Unknown							

RxControl: Communication > Registration

Use these commands to define/inquire the name of the application that is currently using a given connection descriptor (*Cd* - see 1.1.6).

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 + Bluetooth + WiFiAccessPoints all							

RxControl: File > 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, with the following exceptions. After reset, the `Current` and `Boot` configuration files are erased (see the **`exeCopyConfigFile`** command), but the `User1` and `User2` configuration files are kept unchanged.
`Bluetooth`	The list of known Bluetooth devices is erased.
`WiFiAccessPoints`	The list of known WiFi access points is erased (see the **`exeAddWiFiAccessPoint`** command).

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>
```

3.2.2 Tracking and Measurement Generation

sst gst	**setSatelliteTracking** **getSatelliteTracking**	*Satellite*								
		none + G01 ... G32 + R01 ... R24 + E01 ... E32 + S120 ... S140 + J01 + J02 + J03 + GPS + GLONASS + GALILEO + SBAS + QZSS all								

RxControl: 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` and `Jxx` refer to a GPS, Galileo, GLONASS, SBAS or QZSS satellite respectively. GLONASS satellites must be referenced by their slot number (from 1 to 24) in this 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
  +G28+G29+G30+G31+G32+S120+S121+S123+S123+S124+S125+S126+S127+S128
  +S129+S130+S131+S132+S133+S134+S135+S136+S137+S138+S139+S140
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
  +G28+G29+G30+G31+G32+S121+S122+S123+S124+S125+S126+S127+S128+S129
  +S130+S131+S132+S133+S134+S135+S136+S137+S138+S139+S140
COM1>
```

3.2.3 Navigation Filter

sao gao	**setAntennaOffset** **getAntennaOffset**	***Antenna*** *Antenna*	*DeltaE*	*DeltaN*	*DeltaU*					
		+ Main all	-1000.0000 . . . 0.0000 . . . 1000.0000 m	-1000.0000 . . . 0.0000 . . . 1000.0000 m	-1000.0000 . . . 0.0000 . . . 1000.0000 m					

RxControl: 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, see section 2.1.4) 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.

Example

```
COM1>  sao, Main, 0.1, 0.0, 1.3<CR>
$R: sao, Main, 0.1, 0.0, 1.3
  AntennaOffset, Main, 0.1000, 0.0000, 1.3000
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

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

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 and the most recently received differential corrections are extrapolated to the current time.

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

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.

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>
```

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

RxControl: 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'. 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>
```

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

RxControl: 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 combined with the "+" operator. Refer to section 2.1 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.

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>
```

sspc gspc	setStaticPosCartesian getStaticPosCartesian	*Position* *Position*	*X*	*Y*	*Z*	*Datum*				
		+ Cartesian1 + Cartesian2 + Cartesian3 + Cartesian4 + Cartesian5 all	-8000000.0000 . . . 0.0000 . . . 8000000.0000 m	-8000000.0000 . . . 0.0000 . . . 8000000.0000 m	-8000000.0000 . . . 0.0000 . . . 8000000.0000 m	WGS84 ETRS89 NAD83 NAD83_PA NAD83_MA GDA94 Other				

RxControl: 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. The specified datum is reflected in the `Datum` field of the position-related SBF blocks (e.g. `PVTCartesian`). Note that the receiver does not apply any datum transformation to the *X*, *Y* and *Z* coordinates. In particular, the coordinates are encoded without change into the relevant differential correction messages.

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)
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 Other				

RxControl: 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>
```

3.2.4 Session Settings

smp gmp	**setMarkerParameters** **getMarkerParameters**	*MarkerName (60)*	*MarkerNumber (2(*	*MarkerType (20)*						
		SEPT	Unknown	Unknown						

RxControl: 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>
```

sop gop	**setObserverParameters** **getObserverParameters**	*Observer (20)*	*Agency (40)*							
		Unknown	Unknown							

RxControl: 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>
```

3.2.5 General Input/Output

scs gcs	**setCOMSettings** **getCOMSettings**	***Cd*** *Cd*	*Rate*							
		+ COM1 all	baud2400 baud4800 baud9600 baud19200 baud38400 baud57600 baud115200							

RxControl: 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.

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<CR>
$R: scs, COM1, baud19200
  COMSettings, COM1, baud19200
COM1>
```

sdio gdio	**setDataInOut** **getDataInOut**	***Cd*** *Cd*	*Input*	*Output*	*Show*					
		+ DSK1	SBF	none	(off)					
		+ COM1	none	+ RTCMv2	(on)					
		+ USB1	CMD	+ RTCMv3						
		+ USB2	RTCMv2	+ CMRv2						
		+ IP10 ... IP17	RTCMv3	+ SBF						
		+ NTR1	CMRv2	+ NMEA						
		+ IPS1	DC1	+ ASCIIDisplay						
		+ IPS2	DC2	+ DC1						
		+ IPS3	NMEA	+ DC2						
		+ BT01	ASCIIIN							
		+ IPR1	auto							
		+ IPR2								
		+ IPR3								
		all								

RxControl: 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* - see 1.1.6).

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. `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` 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 `ASCIIDisplay` 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 `DCi` to connect the input of pipe *i* to the specified connection. Set the *Output* argument to `DCi` 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, and the *Output* argument is ignored for input-only connections. See section 1.1.6 for details.

Note that not all input connections can accept user commands, check section 1.1.6 for details.

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>
```

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

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. If there is no incoming data for more than 10 seconds, the receiver closes the connection and tries to reconnect.

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.

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.

⚠ When selecting a port number, make sure to avoid conflicts with other services.

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>
```

siss 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

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. Remember to enclose the *UDPAddress* argument between double quotes when it contains whitespaces.

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 *UPDAddress* argument is ignored in TCP mode.

⚠ When selecting a port number, make sure to avoid conflicts with other services.

Set the *Port* argument to 0 to disable an IPS connection.

All IPS connections must use different ports. Set the *Port* argument to 0 to disable an IPS connection.

Example

```
COM1> siss, IPS1, 28785, UDP, 255.255.255.255<CR>
$R: siss, IPS1, 28785, UDP, 255.255.255.255
  IPServerSettings, IPS1, 28785, UDP, 255.255.255.255
COM1>
```

snts gnts	setNtripSettings getNtripSettings	*Cd* *Cd*	*Mode*	*Caster (40)*	*Port*	*UserName (20)*	*Password (40)*	*MountPoint (32)*	*Version*	*SendGGA*
		+ NTR1 all	off Server Client		0 ... 2101 ... 65535				v2	auto off sec1 sec5 sec10 sec60

RxControl: 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	lstNTRIPSourceTable	*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>  lnst, ntripcaster <CR>
$R; lnst, 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>
```

3.2.6 WiFi Settings

eawa gawa	**exeAddWiFiAccessPoint** **getAddWiFiAccessPoint**	***SSID (32)***	***Key (40)***							

RxControl: Communication > WiFi Settings > Client Config

Use this command to add a WiFi access point to the list of known access points, or to modify the password of a known access point. The *SSID* argument is the identifier of the access point and the *Key* argument is the password needed to connect to the access point.

This command must be entered for all access points that the receiver is to connect to. For open access points that do not require a key, set *Key* to "".

When the receiver is configured in WiFi client mode with the **setWiFiMode** command, it will check if a known access point is reachable and automatically connect to it.

By default, if multiple WiFi access points are reachable, the last one that was added with the **exeAddWiFiAccessPoint** command has the priority. Use the **exeManageWiFiAccessPoint** to overrule this and manually define the preferred access point.

The command permanently adds the WiFi access point to the list of known networks. The list is persistent across power cycles. The **exeManageWiFiAccessPoint** command can be used to remove an access point from the list (i.e. to forget an access point), and the **exeResetReceiver, hard, WiFiAccessPoints** command can be used to erase the complete list.

Use the command **lstWiFiAccessPoints** to get a list of the known and/or reachable WiFi access points.

Example

```
COM1>  eawa, MyWiFi, 12345678<CR>
$R: eawa, MyWiFi, 12345678
  AddWiFiAccessPoint, MyWiFi, "7VBC0NULTV6I"
COM1>
```

emwa gmwa	**exeManageWiFiAccessPoint** **getManageWiFiAccessPoint**	***SSID (32)***	***Action***							
			Promote Remove							

RxControl: Communication > WiFi Settings > Client Config

Use this command to remove a WiFi access point from the list of known access points, or to promote it as preferred access point.

If the *Action* argument is `Promote`, the access point identified by the *SSID* argument is given the highest priority in case multiple known access points are reachable.

If the *Action* argument is `Remove`, the access point identified by the *SSID* argument is removed from the list of known access points. This will prevent the receiver from connecting to this access point until it is re-enabled with the **`exeAddWiFiAccessPoint`** command.

Example

```
COM1>  emwa, MyWiFi, Promote<CR>
$R: emwa, MyWiFi, Promote
  ManageWiFiAccessPoint, MyWiFi, Promote
COM1>
```

swfa gwfa	**setWiFiAccessPoint** **getWiFiAccessPoint**	*SSID (32)*	*EncryptionType*	*Key (40)*	*Channel*	*Hotspot*	*SSIDActual (32)*			
		default	none WPA-PSK WPA2	password	1 ... 6 ... 11	off on	model-sn			

RxControl: Communication > WiFi Settings > Acces Point Config

Use this command to configure the WiFi access point.

The SSID is the name of your receiver in the WiFi network. By default, the SSID is the receiver model name followed by its serial number. Use the *SSID* argument to assign your receiver a different WiFi access point name. Using the reserved keyword "default" reverts to the default SSID.

By default, WiFi encryption is turned off. The encryption type and password can be specified with the second and third arguments. Note that the receiver always applies WPA2 even when the *EncryptionType* argument is set to `WPA-PSK`.

The *Channel* argument sets the WiFi frequency channel to be used. It is not necessary to change this value unless you notice interference problems with another nearby WiFi device.

The *Hotspot* argument specifies whether the receiver can be used as a mobile hotspot (wireless tethering). When set to `on`, WiFi clients can access the Internet through the receiver. When enabling the hotspot, beware that the traffic to the Internet may be routed through the internal cellular modem, which may incur data charges.

The last argument (*SSIDActual*) is read-only. It shows the actual SSID. It is a copy of the *SSID* argument except when the *SSID* argument is set to "default".

Example

```
COM1> swfa, , WPA2, password<CR>
$R: swfa, , WPA2, password
  WiFiAccessPoint, default, WPA2, "7VBC0NULTV6I", 6, off, "RxName
     -123456"
COM1>
```

lwa	lstWiFiAccessPoints	*Type*								
		+ Known + Reachable all								

Use this command to list the known and/or the reachable WiFi access points.

The following information is provided for each access points (AP): the SSID, the signal level in dBm (only if AP is in reach), the security type, the current status (`Connected`, `Known` or `Unknown`), and, for known APs, the access point priority (P1 for highest priority). A "known" AP is an AP that has been defined with the **`exeAddWiFiAccessPoint`** command.

The *Type* argument defines the contents of the list:

Type	Description
`Known`	List of known access points.
`Reachable`	List of the access points that are currently in reach of the receiver.

Example

```
COM1>  lwa, all <CR>
$R; lwa, all
---->
$-- BLOCK 1 / 0 C
"TestAP",-62,Unsecured,Unknown,None
"MyAP",-75,Unsecured,Connected,P2
"guest",-78,WPA,Unknown,None
COM1>
```

swfm gwfm	**setWiFiMode** **getWiFiMode**	*Enable*	*Mode*							
		off on	AccessPoint Client							

RxControl: Communication > WiFi Settings > General

Use this command to turn WiFi on and off, and to specify in which WiFi mode the receiver should operate (client or access point).

Note that double-pressing the Altus NR2 button while the receiver is on has the effect of toggling WiFi on and off in turn.

Example

```
COM1>  swfm, off<CR>
$R: swfm, off
  WiFiMode, off, AccessPoint
COM1>
```

3.2.7 Bluetooth Settings

sbtp gbtp	**setBTParameters** **getBTParameters**	*Enable*	*DeviceName (32)*	*PairingCode (8)*	*Discoverable*	*DeviceNameActual*				
		off on	default	1234	off on	model-serialnumber				

RxControl: Communication > Bluetooth Settings

Use this command to configure Bluetooth.

The first argument turns Bluetooth on and off.

By default, the receiver identifies itself by its model name followed by its serial number. Use the *DeviceName* argument to assign your receiver a different Bluetooth name. Using the reserved keyword "default" reverts to the default name.

The Bluetooth pairing code can be specified with the third argument.

The *Discoverable* argument specifies whether other Bluetooth devices are allowed to see your receiver (*Discoverable* set to `on`), or not (*Discoverable* set to `off`).

The fifth argument *DeviceNameActual* is read-only. It shows the actual Bluetooth device name. It is a copy of the *DeviceName* argument except when the *DeviceName* argument is set to "default".

Example

```
COM1> sbtp, off<CR>
$R: sbtp, off
  BluetoothParameters, off, default, 1234, on, "RxName-123456"
COM1>
```

3.2.8 Cellular Modem Settings

scem gcem	**setCellularParameters** **getCellularParameters**	*Power*	*Connect*	*APN (32)*	*User (32)*	*Password (40)*	*Standard*			
		off on	off on				+ 2G + 3G all			

RxControl: Communication > Cellular Settings

Use this command to configure the cellular modem.

The first argument (*Power*) turns the cellular modem on and off, and the second argument (*Connect*) specifies whether the modem should connect to the network or just remain in standby mode.

The *APN* argument specifies the Access Point Name of the network you want to communicate with, and the *User* and *Password* arguments are the optional username and password.

The *Standard* argument can be used to restrict the connection type to 2G-only, 3G-only, or both (default). See the `ConnectionType` field of the `CellularStatus` SBF block for details.

Example

```
COM1> scem, on, apn.com<CR>
$R: scem, on, apn.com
  CellularParameters, on, "apn.com", "", "", 2G+3G
COM1>
```

scep gcep	**setCellularPIN** **getCellularPIN**	*PIN (20)*									

RxControl: Communication > Cellular Settings

Use this command to enter the cellular PIN code. Note that the PIN is obscured in the command reply and in the reply of **getCellularPIN**.

The user can control that the PIN is correct by checking the `ErrorCode` field of the `CellularStatus` SBF block. The SIM card is blocked after entering three wrong PIN codes. To unblock it, you will need to enter the PUK code with the **exeUnblockCellular** command.

The PIN code is kept in non-volatile memory and does not need to be re-entered at each boot of the receiver.

Note that this command is not shown in the output of the **lstConfigFile** command.

Use the **exeChangeCellularPIN** to change the PIN code.

Example

```
COM1>  scep, 1234<CR>
$R: scep, 1234
  CellularPIN, rerdvd
COM1>
```

eccp gccp	**exeChangeCellularPIN** **getChangeCellularPIN**	***OldPIN (20)***	***NewPIN (20)***							

RxControl: Communication > Cellular Settings

Use this command to change the cellular PIN code. The current and new PIN codes must be provided in the first and second arguments.

Note that the PIN codes are obscured in the reply to the command.

Example

```
COM1>  eccp, 1234, 5678<CR>
$R: eccp, 1234, 5678
  ChangeCellularPIN, "F7DFKKKBF7BL0GPP5U2", "F7DFKKKBF7BL0GPP5U2"
COM1>
```

eunc gunc	**exeUnblockCellular** **getUnblockCellular**	***PUK (20)***	***PIN (20)***							

RxControl: Communication > Cellular Settings

Use this command to enter the PUK code to unblock the cellular SIM card.

The SIM card is blocked after entering three wrong PIN codes (see the **setCellularPIN** command).To unblock it, you need to enter the PUK code with this command. At the same time, you will need to provide a new PIN code.

Note that the status of the cellular module is reported in the `CellularStatus` SBF block. The PUK code must be entered when the `ErrorCode` field of that block is set to "SIM card blocked".

Example

```
COM1>  eunc, 12345678, 9876<CR>
$R: eunc, 12345678, 9876
  UnblockCellular, "F7DFKKKBF7BL0GPP5U2", "F7DFKKKBF7BL0GPP5U2"
COM1>
```

3.2.9 NMEA Configuration

enoc gnoc	**exeNMEAOnce** **getNMEAOnce**	***Cd***	***Messages***							
		DSK1 <u>COM1</u> USB1 USB2 IP10 ... IP17 NTR1 IPS1 IPS2 IPS3 BT01	+ GGA + GLL + GRS + GSA + GST + GSV + RMC + VTG + ZDA + LLQ + GGQ + LLK + TXTbase + TFM + SNC + SCL + SBT							

RxControl: Communication > Output Settings > NMEA Output Once

Use this command to output a set of NMEA messages 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 (see 1.1.6) on which the message(s) should be output and the *Messages* argument defines the list of messages that should be output. Refer to appendix C for a short description of the NMEA sentences.

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 + Stream2 all	none DSK1 COM1 USB1 USB2 IP10 ... IP17 NTR1 IPS1 IPS2 IPS3 BT01	none + GGA + GLL + GRS + GSA + GST + GSV + RMC + VTG + ZDA + LLQ + GGQ + LLK + TXTbase + TFM + SNC + SCL + SBT	off OnChange msec10 msec20 msec40 msec50 msec100 msec200 msec500 sec1 sec2 sec5 sec10					

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

Use this command to output a set of NMEA messages on a given connection at a regular interval. The *Cd* argument defines the connection descriptor (see 1.1.6) on which the message(s) should be output and the *Messages* argument defines the list of messages that should be output. Refer to appendix C for a short description of the NMEA sentences.

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

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 gnti	**setNMEATalkerID** **getNMEATalkerID**	*TalkerID*								
		GP GN								

RxControl: 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 gnv	**setNMEAVersion** **getNMEAVersion**	*Version*								
		v3x v4x								

RxControl: 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>
```

3.2.10 SBF Configuration

esoc gsoc	exeSBFOnce getSBFOnce	*Cd*	*Messages*							
		DSK1 COM1 USB1 USB2 IP10 ... IP17 NTR1 IPS1 IPS2 IPS3 BT01	[SBF List] + Measurements + GPS + GLO + GAL + GEO + QZS + PVTCart + PVTGeod + PVTExtra + Attitude + Time + Status + UserGroups + Rinex + Support + RawData + PostProcess + GUI							

RxControl: 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 (see 1.1.6) 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] supported by the **exeSBFOnce** command can be found in appendix B.

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
`Measurements`	+MeasEpoch +MeasExtra +EndOfMeas
`GPS`	+GPSNav +GPSAlm +GPSIon +GPSUtc
`GLO`	+GLONav +GLOAlm +GLOTime
`GAL`	+GALNav +GALAlm +GALIon +GALUtc +GALGstGps
`GEO`	+GEONav +GEOAlm
`QZS`	+QZSNav
`PVTCart`	+PVTCartesian +PosCovCartesian +VelCovCartesian +BaseVectorCart
`PVTGeod`	+PVTGeodetic +PosCovGeodetic +VelCovGeodetic +BaseVectorGeod +PosLocal
`PVTExtra`	+DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +GEOCorrections +BaseLine +PVTSupport +EndOfPVT

Messages (Continued)	Description
`Attitude`	+AttEuler +AttCovEuler +EndOfAtt
`Time`	+ReceiverTime
`Status`	+SatVisibility +ChannelStatus +ReceiverStatus +InputLink +OutputLink +IPStatus +NTRIPClientStatus +WiFiAPStatus +WiFiClientStatus +CellularStatus +BluetoothStatus +BatteryStatus +QualityInd +DiskStatus
`UserGroups`	+Group1 +Group2 +Group3 +Group4
`Rinex`	+MeasEpoch +GPSNav +GPSIon +GPSUtc +GLONav +GALNav +GALUtc +GALGstGps +GEONav +QZSNav +PVTGeodetic +ReceiverSetup +Comment
`Support`	+MeasEpoch +MeasExtra +EndOfMeas +GPSNav +GPSAlm +GPSIon +GPSUtc +GLONav +GLOAlm +GLOTime +GALNav +GALAlm +GALIon +GALUtc +GALGstGps +GEONav +GEOAlm +QZSNav +PVTGeodetic +PosCovGeodetic +BaseVectorGeod +AttEuler +DOP +PVTSupport +EndOfPVT +ChannelStatus +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +RxComponents +Commands +IPStatus +NTRIPClientStatus +WiFiAPStatus +WiFiClientStatus +CellularStatus +BluetoothStatus +BatteryStatus +QualityInd +DiskStatus
`RawData`	+MeasEpoch +MeasExtra +GPSNav +GLONav +GALNav +GEONav +QZSNav +PVTGeodetic +ReceiverSetup +Commands +Comment
`PostProcess`	+MeasEpoch +MeasExtra +GPSNav +GPSIon +GPSUtc +GLONav +GLOTime +GALNav +GALIon +GALUtc +GALGstGps +GEONav +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 <CR>
$R: esoc, COM1, MeasEpoch
  SBFOnce, COM1, MeasEpoch
COM1>
```

sso gso	**setSBFOutput** **getSBFOutput**	***Stream*** *Stream*	*Cd*	*Messages*	*Interval*					
		+ Stream1 ... Stream8 + Res1 + Res2 + Res3 + Res4 all	none DSK1 COM1 USB1 USB2 IP10 ... IP17 NTR1 IPS1 IPS2 IPS3 BT01	none [SBF List] + Measurements + RawNavBits + GPS + GLO + GAL + GEO + QZS + PVTCart + PVTGeod + PVTExtra + Attitude + Time + DiffCorr + Status + 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

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* - see 1.1.6). 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] can be found in appendix B.

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 +EndOfMeas
`RawNavBits`	+GPSRawCA +GPSRawL2C +GLORawCA +GALRawINAV +GEORawL1 +QZSRawL1CA +QZSRawL2C
`GPS`	+GPSNav +GPSAlm +GPSIon +GPSUtc
`GLO`	+GLONav +GLOAlm +GLOTime
`GAL`	+GALNav +GALAlm +GALIon +GALUtc +GALGstGps +GALSARRLM
`GEO`	+GEOMT00 +GEOPRNMask +GEOFastCorr +GEOIntegrity +GEOFastCorrDegr +GEONav +GEODegrFactors +GEONetworkTime +GEOAlm +GEOIGPMask +GEOLongTermCorr +GEOIonoDelay +GEOServiceLevel +GEOClockEphCovMatrix
`QZS`	+QZSNav
`PVTCart`	+PVTCartesian +PosCovCartesian +VelCovCartesian +BaseVectorCart
`PVTGeod`	+PVTGeodetic +PosCovGeodetic +VelCovGeodetic +BaseVectorGeod +PosLocal

Messages (Continued)	Description
`PVTExtra`	+DOP +PVTSatCartesian +PVTResiduals +RAIMStatistics +GEOCorrections +BaseLine +PVTSupport +EndOfPVT
`Attitude`	+AttEuler +AttCovEuler +EndOfAtt
`Time`	+ReceiverTime
`DiffCorr`	+DiffCorrIn +BaseStation +RTCMDatum
`Status`	+SatVisibility +ChannelStatus +ReceiverStatus +InputLink +OutputLink +IPStatus +NTRIPClientStatus +WiFiAPStatus +WiFiClientStatus +CellularStatus +BluetoothStatus +BatteryStatus +QualityInd +DiskStatus
`UserGroups`	+Group1 +Group2 +Group3 +Group4
`Rinex`	+MeasEpoch +GEORawL1 +GPSNav +GPSIon +GPSUtc +GLONav +GALNav +GALUtc +GALGstGps +GEONav +QZSNav +PVTGeodetic +ReceiverSetup +Comment
`Support`	+MeasEpoch +MeasExtra +EndOfMeas +GPSRawCA +GPSRawL2C +GLORawCA +GALRawINAV +GEORawL1 +QZSRawL1CA +QZSRawL2C +GPSNav +GPSAlm +GPSIon +GPSUtc +GLONav +GLOAlm +GLOTime +GALNav +GALAlm +GALIon +GALUtc +GALGstGps +GEONav +GEOAlm +QZSNav +PVTGeodetic +PosCovGeodetic +BaseVectorGeod +AttEuler +DOP +PVTSupport +EndOfPVT +DiffCorrIn +BaseStation +ChannelStatus +ReceiverStatus +InputLink +OutputLink +ReceiverSetup +RxComponents +Commands +IPStatus +NTRIPClientStatus +WiFiAPStatus +WiFiClientStatus +CellularStatus +BluetoothStatus +BatteryStatus +QualityInd +DiskStatus
`RawData`	+MeasEpoch +MeasExtra +GPSRawCA +GPSRawL2C +GLORawCA +GALRawINAV +GEORawL1 +QZSRawL1CA +QZSRawL2C +GPSNav +GLONav +GALNav +GEONav +QZSNav +PVTGeodetic +DiffCorrIn +ReceiverSetup +Commands +Comment
`PostProcess`	+MeasEpoch +MeasExtra +GEORawL1 +GPSNav +GPSIon +GPSUtc +GLONav +GLOTime +GALNav +GALIon +GALUtc +GALGstGps +GALSARRLM +GEONav +QZSNav +DiffCorrIn +ReceiverSetup +Commands
`GUI`	+MeasEpoch +EndOfMeas +GEOIGPMask +GEOIonoDelay +EndOfPVT +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 (see section 4.1.8). 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 the value of the *Interval* argument and always assumes `OnChange`. The list of those blocks can be found in appendix B (see the "Flex Rate" column).

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>
```

3.2.11 RTCM v2.x Settings

sr2f gr2f	**setRTCMv2Formatting** **getRTCMv2Formatting**	*ReferenceID*	*GLOToD*								
		0 ... 1023	Tk Tb								

RxControl: 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

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*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.

Refer to appendix D for an overview of the supported RTCM v2.x messages.

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

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 RCTCM 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 gr2m	**setRTCMv2Message16** **getRTCMv2Message16**	*Message (90)*								
		Unknown								

RxControl: 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 + USB1 + USB2 + IP10 ... IP17 + NTR1 + IPS1 + IPS2 + IPS3 + BT01 all	none + RTCM1 + RTCM3 + RTCM9 + RTCM16 + RTCM18|19 + RTCM20|21 + RTCM22 + RTCM23|24 + RTCM31 + RTCM32 + RTCM17 + DGPS + RTK all							

RxControl: 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* - see 1.1.6). 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`".

Refer to appendix D for an overview of the supported RTCM v2.x messages.

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>
```

3.2.12 RTCM v3.x Settings

sr3d gr3d	**setRTCMv3Delay** **getRTCMv3Delay**	*Delay*								
		0 . . . 600 sec								

RxControl: 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 + GLOL1CA + GLOL2CA + GALL1BC + QZSL1CA + QZSL2C all	L2CA L2P						

RxControl: 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

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`. Refer to appendix D for an overview of the supported RTCM v3.x messages.

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 gr3m	**setRTCMv3Message1029** **getRTCMv3Message1029**	*Message (120)*								
		Unknown								

RxControl: 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 + USB1 + USB2 + IP10 ... IP17 + NTR1 + IPS1 + IPS2 + IPS3 + BT01 all	none + RTCM1001 + RTCM1002 + RTCM1003 + RTCM1004 + RTCM1005 + RTCM1006 + RTCM1007 + RTCM1008 + RTCM1009 + RTCM1010 + RTCM1011 + RTCM1012 + RTCM1013 + RTCM1019 + RTCM1020 + RTCM1029 + RTCM1033 + RTCM1044 + RTCM1045 + RTCM1071 ... RTCM1077 + RTCM1081 ... RTCM1087 + RTCM1091 ... RTCM1097 + RTCM1111 ... RTCM1117 + MSM1 + MSM2 + MSM3 + MSM4 + MSM5 + MSM6 + MSM7 all							

RxControl: 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* - see 1.1.6). The *Messages* argument specifies the RTCM message types to be enabled.

A short description of the supported RTCM v3.x message types can be found in appendix D. `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.

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>
```

3.2.13 CMR v2.0 Settings

sc2f gc2f	**setCMRv2Formatting** **getCMRv2Formatting**	*ReferenceID*								
		0 . . . 31								

RxControl: 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

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.

Refer to appendix D for an overview 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 gc2m	**setCMRv2Message2** **getCMRv2Message2**	*ShortID (8)*	*LongID (50)*	*COGO (16)*						
		Unknown	Unknown	Unknown						

RxControl: 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.

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 + USB1 + USB2 + IP10 ... IP17 + NTR1 + IPS1 + IPS2 + IPS3 + BT01 all	none + CMR0 + CMR1 + CMR2 + CMR3 all							

RxControl: 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* - see 1.1.6). The *Messages* argument specifies the CMR message types to be enabled. Refer to appendix D for an overview 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>
```

3.2.14 Internal Disk Logging

sdfa gdfa	**setDiskFullAction** **getDiskFullAction**	***Disk*** *Disk*	*Action*							
		+ DSK1 all	DeleteOldest StopLogging							

RxControl: 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.

Examples

```
COM1> sdfa, DSK1, StopLogging <CR>
$R: sdfa, DSK1, StopLogging
  DiskFullAction, DSK1, StopLogging
COM1>

COM1> gdfa <CR>
$R: gdfa
  DiskFullAction, DSK1, StopLogging
COM1>
```

sfn gfn	**setFileNaming** **getFileNaming**	***Disk*** *Disk*	*NamingType*	*FileName (8)*						
		+ DSK1 all	FileName Incremental IGS15M IGS1H IGS6H IGS24H	log						

RxControl: 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.

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>
```

emd gmd	**exeManageDisk** **getManageDisk**	***Disk***	***Action***							
		DSK1	Unmount Format							

RxControl: 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.

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>
```

erf grf	**exeRemoveFile** **getRemoveFile**	***Disk***	***FileName (60)***							
		DSK1	none all							

RxControl: 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>
```

Chapter 4

SBF Reference

4.1 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.

4.1.1 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 4.1.6).
`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.

4.1.2 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 list of SBF blocks available on your receiver can be found in Appendix B.

4.1.3 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 4.1.7). 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 4.1.8). 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.

4.1.4 Sub-blocks

Some blocks contain sub-blocks. For example, the `PVTSatCartesian` block contains N `SatPos` 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.

4.1.5 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.

4.1.6 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 4.1.1, 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 4.1.12). 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 4.2, 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.

4.1.7 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.

4.1.8 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 4.2. To instruct the receiver to output a given block at its default rate, the "OnChange" rate has to be specified in the **setSBFOutput** command.

Some blocks can only be output at their default rate (e.g. the GPSNav block). Others can be decimated to a user-selectable 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 ("Flex Rate") and whether it belongs to the "output once" set is indicated in the SBF block list in Appendix B.

Attempting to force another rate than the default one for those blocks that do not support "Flex Rate" has no effect: those blocks are always output at their default rate.

4.1.9 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 Value	Description	RINEX satellite code
SVID or PRN	u1	0	Satellite ID: The following ranges are defined:	
			1-37: PRN number of a GPS satellite	*Gnn* (*nn* = SVID)
			38-61: Slot number of a GLONASS satellite with an offset of 37	*Rnn* (*nn* = SVID-37)
			62: GLONASS satellite of which the slot number is not known	NA
			71-102: PRN number of a GALILEO satellite with an offset of 70	*Enn* (*nn* = SVID-70)
			107-119: L-Band (MSS) satellite. Corresponding satellite name can be found in the LBandBeams block.	NA
			120-140: PRN number of an SBAS satellite	*Snn* (*nn* = SVID-100)
			141-172: PRN number of a Compass/BeiDou satellite with an offset of 140	*Cnn* (*nn* = SVID-140)
			181-187: PRN number of a QZSS satellite with an offset of 180	*Jnn* (*nn* = SVID-180)
			191-197: PRN number of an IRNSS satellite with an offset of 190	*Inn* (*nn* = SVID-190)
FreqNr	u1	0	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). For non-GLONASS satellites, FreqNr is reserved and must be ignored by the decoding software.	

4.1.10 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+(FreqNr-8)*9/16, with FreqNr as defined in section 4.1.9.	1C
9	Reserved		
10	GLO_L2-P	1246.00+(FreqNr-8)*7/16	2P
11	GLO_L2-CA	1246.00+(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		

4.1.11 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.

4.1.12 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.

4.2 SBF Block Definitions

4.2.1 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N1	u1			Number of MeasEpochChannelType1 sub-blocks in this MeasEpoch block.
SB1Length	u1	1 byte		Length of a MeasEpochChannelType1 sub-block, excluding the nested MeasEpochChannelType2 sub-blocks

SB2Length	u1	1 byte		Length of a MeasEpochChannelType2 sub-block
CommonFlags	u1			Bit field containing flags common to all measurements. Bit 0: Multipath mitigation: if this bit is set, multipath mitigation is enabled. (see the **setMultipathMitigation** command). Bit 1: Smoothing of code: if this bit is set, at least one of the code measurements are smoothed values (see **setSmoothingInterval** command). 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. Bit 3: Clock steering: this bit is set if clock steering is active (see **setClockSyncThreshold** command). Bit 4: Not applicable. Bits 5-6: Reserved Bit 7: Scrambling: bit set when the measurements are scrambled. Scrambling is applied when the "Measurement Availability" permission is not granted (see the **lif,Permissions** command).
Rev 1 CumClkJumps	u1	0.001 s		Cumulative millisecond clock jumps since start-up, with an ambiguity of k*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	...	...		*A succession of N1 MeasEpochChannelType1 sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

MeasEpochChannelType1 sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
RxChannel	u1			Receiver channel on which this satellite is currently tracked (see 4.1.11).
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number, see 4.1.10. Bits 5-7: Antenna ID: 0 for main, 1 for *Aux1* and 2 for *Aux2*
SVID	u1			Satellite ID, see 4.1.9
Misc	u1	4294967.296 m	0 (1)	Bit field containing the MSB of the pseudorange. Bits 0-3: CodeMSB: MSB of the pseudorange (this is an unsigned value). Bits 4-7: Reserved
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

(1) The pseudorange is invalid if both CodeMSB is 0 and CodeLSB is 0.
(2) The carrier phase is invalid if both CarrierMSB is -128 and CarrierLSB is 0.

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]/λ +(CarrierMSB*65536+CarrierLSB)*0.001 where λ is the carrier wavelength corresponding to the frequency of the signal type in the Type field above: λ=299792458/f_L m, with f_L the carrier frequency as listed in section 4.1.10.
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 4.1.9), 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 4.1.5
Type2	...	...		*A succession of N2 MeasEpochChannelType2 sub-blocks, see definition below*

MeasEpochChannelType2 sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number, see 4.1.10. Bits 5-7: Antenna ID: 0 for main, 1 for *Aux1* and 2 for *Aux2*
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.
CN0	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 pseudo-range, 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]/λ +(CarrierMSB*65536+CarrierLSB)*0.001 where λ is the carrier wavelength corresponding to the signal type in the Type field.
DopplerOffsetLSB	u2	0.0001 Hz	0 (4)	LSB of the Doppler offset relative to the Doppler in the MeasEpochChannelType1 sub-block. To compute the Doppler, use: D_{type2}[Hz] = D_{type1}[Hz]*α +(DopplerOffsetMSB*65536+DopplerOffsetLSB) *1e-4, 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 4.1.10 for a list of all carrier frequencies).
Padding	u1[..]			Padding bytes, see 4.1.5

(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.

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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: $\sigma^2_{\mathrm{Doppler}}$[$mHz^2$] = CarrierVariance * DopplerVarFactor, Where CarrierVariance can be found for each measurement type in the MeasExtraChannelSub sub-blocks.
ChannelSub	...	...		*A succession of N MeasExtraChannelSub sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

`MeasExtraChannelSub` sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
RxChannel	u1			Receiver channel on which this satellite is currently tracked (see 4.1.11).
Type	u1			Bit field indicating the signal type and antenna ID: Bits 0-4: signal number, see 4.1.10. Bits 5-7: Antenna ID: 0 for main, 1 for *Aux1* and 2 for *Aux2*
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 `DopplerVarFactor` 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.
Rev 1 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.
Rev 2 Info	u1			Bit field: Bits 0-3: Reserved Bits 4-7: Reserved.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.2 Navigation Page Blocks

GPSRawCA	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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			NAVBits contains the 300 bits of a GPS C/A subframe. 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: 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}). Bits 6-29: source data bits (referred to as d_n in the GPS ICD). The first received bit is the MSB. Bits 30-31: Reserved
Padding	u1[..]			Padding bytes, see 4.1.5

GPSRawL2C	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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			NAVBits contains the 300 bits of a GPS CNAV subframe. 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.
Padding	u1[..]			Padding bytes, see 4.1.5

GLORawCA	Number: 4026 "OnChange" interval: 2s

This block contains the 85 bits of a GLONASS L1CA or L2CA navigation string.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 Bits 5-7: Reserved
FreqNr	u1			Frequency number, with an offset of 8. See 4.1.9
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[3]			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). 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.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 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. Bits 6-7: Reserved
FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			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. 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.
Padding	u1[..]			Padding bytes, see 4.1.5

GEORawL1	Number: 4020 "OnChange" interval: 1s

This block contains the 250 bits of a SBAS L1 navigation frame, after Viterbi decoding.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 Bits 5-7: Reserved
FreqNr	u1			Not applicable
Reserved	u1			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[8]			NAVBits contains the 250 bits of a SBAS navigation frame. 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.
Padding	u1[..]			Padding bytes, see 4.1.5

QZSRawL1CA	Number: 4066
	"OnChange" interval: 6s

This block contains the 300 bits of a QZSS C/A subframe.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 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 4.1.5

QZSRawL2C	Number: 4067
	"OnChange" interval: 12s

This block contains the 300 bits of a QZSS L2C CNAV subframe.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10 Bits 5-7: Reserved
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software.
NAVBits	u4[10]			NAVBits contains the 300 bits of a QZSS CNAV subframe. 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.
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.3 GPS Decoded Message Blocks

GPSNav	Number:	5891
	"OnChange" interval:	block generated each time a new navigation data set is received from a GPS satellite

The GPSNav 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the GPS satellite of which the ephemeris is given in this block (see 4.1.9)
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 4.1.5

GPSAlm	Number:	5892
	"OnChange" interval:	block generated each time a new almanac data set is received from a GPS satellite

The GPSAlm 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the GPS satellite of which the almanac is given is this block (see 4.1.9)
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 4.1.5

GPSIon	Number:	5893
	"OnChange" interval:	block generated each time subframe 4, page 18, is received from a GPS satellite

The GPSIon 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the GPS satellite from which the coefficients have been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the GPS satellite from which these UTC parameters have been received (see 4.1.9)
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 4.1.5

4.2.4 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			ID of the GLONASS satellite for which ephemeris is provided in this block (see 4.1.9).
FreqNr	u1			Frequency number of the GLONASS satellite for which ephemeris is provided in this block (see 4.1.9).
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			ID of the GLONASS satellite for which almanac is provided in this block (see 4.1.9).
FreqNr	u1			Frequency number of the GLONASS satellite for which almanac is provided in this block (see 4.1.9). 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		$\Delta\ i_n^A$: 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		$\Delta\ T_n^A$: 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 4.1.5

GLOTime	Number:	4036
	"OnChange" interval:	block generated at the end of each GLONASS superframe, 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			ID of the GLONASS satellite from which the data in this block has been decoded (see 4.1.9).
FreqNr	u1			Frequency number of the GLONASS satellite from which the data in this block has been decoded (see 4.1.9).
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 4.1.5

4.2.5 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			SVID of the Galileo satellite (see 4.1.9)

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 Source field indicates which clock model t_oc refers to.
a_f2	f4	1 s / s^2		SV clock aging. The Source field indicates which clock model a_f2 refers to.
a_f1	f4	1 s / s		SV clock drift. The Source field indicates which clock model a_f1 refers to.
a_f0	f8	1 s		SV clock bias. The Source 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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			SVID of the Galileo satellite from which these almanac parameters have been received (see 4.1.9)
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 4.1.9 for the SVID numbering convention).

health	u2			Bit field indicating the health status (HS) of the E5a, E5b, L1-B, L1-A and E6-A signals: Bit 0: If set, bits 1 and 2 are valid, otherwise they must be ignored. Bits 1-2: 2-bit L1-B$_{\mathrm{HS}}$ Bit 3: If set, bits 4 and 5 are valid, otherwise they must be ignored. Bits 4-5: 2-bit E5b$_{\mathrm{HS}}$ Bit 6: If set, bits 7 and 8 are valid, otherwise they must be ignored. Bits 7-8: 2-bit E5a$_{\mathrm{HS}}$ Bit 9: If set, bits 10 and 11 are valid, otherwise they must be ignored. Bits 10-11: 2-bit L1-A$_{\mathrm{HS}}$ Bit 12: If set, bits 13 and 14 are valid, otherwise they must be ignored. Bits 13-14: 2-bit E6-A$_{\mathrm{HS}}$ Bit 15: Reserved
IODa	u1			4-bit Issue of Data for the almanac.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 4.1.9)
Source	u1			Message type from which the data has been decoded: 2: I/NAV 16: F/NAV
a_i0	f4	$1 \cdot 10^{-22}$ W / (m^2 Hz)		Effective ionization level, a_{i0}
a_i1	f4	$1 \cdot 10^{-22}$ W / (m^2 Hz) / deg		Effective ionization level, a_{i1}
a_i2	f4	$1 \cdot 10^{-22}$ W / (m^2 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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
SVID	u1			SVID of the Galileo satellite from which these parameters have been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
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. RLMLength 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*]			Bits in the RLM message, with the first bit being the MSB of RLMBits[0]. *N* is 3 for a short message (i.e. if RLMLength is 80), and 5 for a long message (i.e. if RLMLength is 160). The 16 unused bits of a short message are set to 0. These bits correspond to the 16 LSBs of RLMBits[2].
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.6 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the QZSS satellite of which the ephemeris is given in this block (see 4.1.9)
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 4.1.5

4.2.7 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
IODP	u1			Issue of data - PRN.
NbrPRNs	u1			Number of PRNs designated in the mask.
PRNMask	u1[NbrPRNs]			List of the PRNs in the PRN mask. 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...
Padding	u1[..]			Padding bytes, see 4.1.5

GEOFastCorr	Number:	5927
	"OnChange" interval:	block generated each time MT02, MT03, MT04, MT05, MT24 and possibly MT00 is received from an SBAS satellite

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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 FastCorr sub-blocks. N depends on the message type as follows: Message type / N: MT00, MT02, MT03, MT04: 13 MT05: 12 MT24: 6
SBLength	u1			Length of the FastCorr sub-blocks in bytes
FastCorr	...	...		*A succession of N FastCorr sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

FastCorr sub-block definition:

Parameter	Type	Units	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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite of which the navigation data is provided here (see 4.1.9)
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 t0
Xgd	f8	1 m / s		X velocity at time-of-day t0
Ygd	f8	1 m / s		Y velocity at time-of-day t0
Zgd	f8	1 m / s		Z velocity at time-of-day t0
Xgdd	f8	1 m / s^2		X acceleration at time-of-day t0
Ygdd	f8	1 m / s^2		Y acceleration at time-of-day t0
Zgdd	f8	1 m / s^2		Z acceleration at time-of-day t0
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which this Network Time data was received (see 4.1.9)
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 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
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite of which the almanac is provided here (see 4.1.9)
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 t0
Yg	f8	1 m		Y position at time-of-day t0
Zg	f8	1 m		Z position at time-of-day t0
Xgd	f8	1 m / s		X velocity at time-of-day t0
Ygd	f8	1 m / s		Y velocity at time-of-day t0
Zgd	f8	1 m / s		Z velocity at time-of-day t0
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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]			List of the IGPs in the IGP mask. 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...
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
N	u1			Number of long-term corrections in this message. This is the number of LTCorr sub-blocks. N can be 0, 1, 2, 3 or 4.
SBLength	u1	1 byte		Length of the LTCorr sub-blocks in bytes
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
LTCorr	...	...		*A succession of N LTCorr sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

LTCorr sub-block definition:

Parameter	Type	Units	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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which the message has been received (see 4.1.9)
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 IDC sub-blocks. N is always 15.
SBLength	u1	1 byte		Length of the IDC sub-blocks in bytes.
Reserved	u1			Reserved for future use, to be ignored by decoding software
IDC	...	...		*A succession of N IDC sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

IDC sub-block definition:

Parameter	Type	Units	Description
IGPMaskNo	u1		Sequence number in the IGP mask (see GEOIGPMask 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 4.1.5

GEOServiceLevel	Number:	5917
	"OnChange" interval:	block generated each time MT27 is received from an SBAS satellite

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			ID of the SBAS satellite from which this service level message was received (see 4.1.9)
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			δUDRE Indicator for users inside the service region, from 0 to 15
dUDREI_Out	u1			δUDRE Indicator for users outside the service region, from 0 to 15
N	u1			Number of Regions in this message. This is the number of ServiceRegion sub-blocks. Ranging from 0 to 7
SBLength	u1	1 byte		Length of the ServiceRegion sub-blocks in bytes
Regions	...	...		*A succession of N ServiceRegion sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

ServiceRegion sub-block definition:

Parameter	Type	Units	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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	SIS time stamp, see 4.1.3
WNc	u2	1 week	65535	
PRN	u1			
IODP	u1			Issue of data - PRN.
N	u1			Number of covariance matrices in this message. This is the number of CovMatrix sub-blocks. N can be 1 or 2.
SBLength	u1	1 byte		Length of the CovMatrix sub-blocks in bytes
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software
CovMatrix	...	...		*A succession of N CovMatrix sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

CovMatrix sub-block definition:

Parameter	Type	Units	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 4.1.5

4.2.8 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 co-ordinate 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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.
X	f8	1 m	$-2 \cdot 10^{10}$	Marker X coordinate in coordinate frame specified by Datum
Y	f8	1 m	$-2 \cdot 10^{10}$	Marker Y coordinate in coordinate frame specified by Datum
Z	f8	1 m	$-2 \cdot 10^{10}$	Marker Z coordinate in coordinate frame specified by Datum
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 TimeSystem 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 4.1.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.

AlertFlag	u1		0	Bit field indicating integrity related information: Bits 0-1: RAIM integrity flag: 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm) Bit 3: Reserved 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. Bits 5-7: Reserved
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	Bit field containing PPP-related information: 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). Bit 12: Reserved Bits 13-15: Type of last seed: 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.
Misc	u1			Bit field containing miscellaneous flags: 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. Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown. Bit 2: Proprietary. Bit 3: Proprietary. Bits 4-5: Proprietary. Bits 6-7: Reserved
Padding	u1[..]			Padding bytes, see 4.1.5

Rev 1 (NrBases, PPPInfo)

Rev 2 (Latency, HAccuracy, VAccuracy, Misc)

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 co-ordinate 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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.
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 Datum)
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 TimeSystem 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 4.1.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.

AlertFlag	u1		0	Bit field indicating integrity related information: Bits 0-1: RAIM integrity flag: 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm) Bit 3: Reserved 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. Bits 5-7: Reserved
NrBases	u1		0	Number of base stations used in the PVT computation.
PPPInfo	u2	1 s	0	Bit field containing PPP-related information: 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). Bit 12: Reserved Bits 13-15: Type of last seed: 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.
Misc	u1			Bit field containing miscellaneous flags: 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. Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown. Bit 2: Proprietary. Bit 3: Proprietary. Bits 4-5: Proprietary. Bits 6-7: Reserved
Padding	u1[..]			Padding bytes, see 4.1.5

Rev 1

Rev 2

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/N_0, 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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	$-2 \cdot 10^{10}$	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_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 4.1.5

`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 m^2 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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_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 4.1.5

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:

$$\begin{pmatrix} \sigma^2_{v_x} & \sigma_{v_x v_y} & \sigma_{v_x v_z} & \sigma_{v_x d} \\ \sigma_{v_y v_x} & \sigma^2_{v_y} & \sigma_{v_y v_z} & \sigma_{v_y d} \\ \sigma_{v_z v_x} & \sigma_{v_z v_y} & \sigma^2_{v_z} & \sigma_{v_z d} \\ \sigma_{d v_x} & \sigma_{d v_y} & \sigma_{d v_z} & \sigma^2_{d} \end{pmatrix}$$

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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_VxVx	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the x-velocity estimate
Cov_VyVy	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the y-velocity estimate
Cov_VzVz	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the z-velocity estimate
Cov_DtDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the clock drift estimate
Cov_VxVy	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the x- and y-velocity estimates
Cov_VxVz	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the x- and z-velocity estimates
Cov_VxDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the x-velocity and the clock drift estimates
Cov_VyVz	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the y- and z-velocity estimates

Cov_VyDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the y-velocity and the clock drift estimates
Cov_VzDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the z-velocity and the clock drift estimates
Padding	u1[..]			Padding bytes, see 4.1.5

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:

$$\begin{pmatrix} \sigma^2_{v_N} & \sigma_{v_N v_E} & \sigma_{v_N v_U} & \sigma_{v_N d} \\ \sigma_{v_E v_N} & \sigma^2_{v_E} & \sigma_{v_E v_U} & \sigma_{v_E d} \\ \sigma_{v_U v_N} & \sigma_{v_U v_E} & \sigma^2_{v_U} & \sigma_{v_U d} \\ \sigma_{d v_N} & \sigma_{d v_E} & \sigma_{d v_U} & \sigma^2_{d} \end{pmatrix}$$

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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_VnVn	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the north-velocity estimate
Cov_VeVe	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the east-velocity estimate
Cov_VuVu	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the up-velocity estimate
Cov_DtDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Variance of the clock drift estimate
Cov_VnVe	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the north- and east-velocity estimates
Cov_VnVu	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the north- and up-velocity estimates
Cov_VnDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the north-velocity and clock drift estimates
Cov_VeVu	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the east- and up-velocity estimates

Cov_VeDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the east-velocity and clock drift estimates
Cov_VuDt	f4	1 m^2 / s^2	$-2 \cdot 10^{10}$	Covariance between the up-velocity and clock drift estimates
Padding	u1[..]			Padding bytes, see 4.1.5

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:

$$\text{Position Dilution of Precision:} \quad PDOP = \sqrt{\mathbf{Q}_{xx} + \mathbf{Q}_{yy} + \mathbf{Q}_{zz}}$$
$$\text{Time Dilution of Precision:} \quad TDOP = \sqrt{\mathbf{Q}_{bb}}$$
$$\text{Horizontal Dilution of Precision:} \quad HDOP = \sqrt{\mathbf{Q}_{\lambda\lambda} + \mathbf{Q}_{\phi\phi}}$$
$$\text{Vertical Dilution of Precision:} \quad VDOP = \sqrt{\mathbf{Q}_{hh}}$$

In these equations, the matrix **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^tA). 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 xDOP 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 4.1.5

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 section 2.1.4 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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.
X	f8	1 m	$-2 \cdot 10^{10}$	Marker X coordinate in coordinate frame specified by Datum
Y	f8	1 m	$-2 \cdot 10^{10}$	Marker Y coordinate in coordinate frame specified by Datum
Z	f8	1 m	$-2 \cdot 10^{10}$	Marker Z coordinate in coordinate frame specified by Datum
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			Bit field containing miscellaneous flags: 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. Bit 1: In RTK mode, set if the phase center variation is compensated for at the rover, unset if not or unknown. Bit 2: Proprietary. Bit 3: Proprietary. Bits 4-5: Proprietary. Bits 6-7: Reserved
Reserved	u1			Reserved for future use.
AlertFlag	u1		0	Bit field indicating integrity related information: Bits 0-1: RAIM integrity flag: 0: RAIM not active (integrity not monitored) 1: RAIM integrity test successful 2: RAIM integrity test failed 3: Reserved Bit 2: set if integrity has failed as per Galileo HPCA (HMI Probability Computation Algorithm) Bit 3: Reserved 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. Bits 5-7: Reserved
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 4.1.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.
Padding	u1[..]			Padding bytes, see 4.1.5

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 **??**.

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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 17: Datum transformation parameters unknown Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.
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 HeightType field of the corresponding RTCMDatum 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 RTCMDatum block having a matching Datum field.
Padding	u1[..]			Padding bytes, see 4.1.5

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 **??**.

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. 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 17: Datum transformation parameters unknown Note: if this field has a non-zero value, all following fields are set to their Do-Not-Use value.
Northing	f8	1 m	$-2\cdot10^{10}$	Marker northing coordinate in the plane grid representation.
Easting	f8	1 m	$-2\cdot10^{10}$	Marker easting coordinate in the plane grid representation.
Alt	f8	1 m	$-2\cdot10^{10}$	Marker height. If the Datum field is in the 20 to 24 range (it is always the case in the current firmware), see the HeightType field of the corresponding RTCMDatum 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 RTCMDatum block having a matching Datum field.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of satellites for which satellite position is provided in this SBF block, i.e. number of SatPos sub-blocks. If N is 0, there are no satellite positions available for this epoch.
SBLength	u1	1 byte		Length of one sub-block
SatPos	...	...		*A succession of N SatPos sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

SatPos sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
SVID	u1			Satellite ID, see 4.1.9
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 (6)	Total slant ionospheric delay at the L1 carrier frequency (1575.42MHz), with a decimeter resolution.
TropoMSB	i2	1 dm	−32768 (7)	Total slant tropospheric delay, with a decimeter resolution.

(6) The ionospheric delay should not be used when IonoMSB is -32768.
(7) The tropospheric delay should not be used when TropoMSB is -32768.

Rev 1

IonoLSB	u1	1.0/256.0 dm	0 (6)	Sub-decimeter part of the slant ionospheric delay. The high-resolution ionospheric delay, expressed in meters, can be computed as: IonoDelay[m] = 0.1*(IonoMSB + IonoLSB/256)
TropoLSB	u1	1.0/256.0 dm	0 (7)	Sub-decimeter part of the slant tropospheric delay. The high-resolution tropospheric delay, expressed in meters, can be computed as: TropoDelay[m] = 0.1*(TropoMSB + 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 4.1.5

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 section **??**):

$$\sigma_{e_i} = \frac{|e_i|}{w_i} \text{ and } \sigma_{y_i} = \sqrt{\frac{MDB}{\sqrt{\lambda_0}}.\sigma_{e_i}}$$

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of satellite/signal pairs for which residual blocks are provided in this SBF block, i.e. number of SatSignalInfo sub-blocks. If N is 0, there are no satellite residuals available for this epoch.
SB1Length	u1	1 byte		Length of a SatSignalInfo sub-block, excluding the nested ResidualInfoCode, ResidualInfoPhase and ResidualInfoDoppler sub-blocks
SB2Length	u1	1 byte		Length of a ResidualInfoCode, ResidualInfoPhase and ResidualInfoDoppler sub-block
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
Residuals	...	...		*A succession of N SatSignalInfo sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

`SatSignalInfo` sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
SVID	u1			Satellite ID, see 4.1.9
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 4.1.10. 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 4.1.9. 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 4.1.9. 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.
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 4.1.5
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

Rev 1

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	Do-Not-Use	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 4.1.5

ResidualInfoPhase sub-block definition:

Parameter	Type	Units	Do-Not-Use	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 4.1.5

ResidualInfoDoppler sub-block definition:

Parameter	Type	Units	Do-Not-Use	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 4.1.5

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 also refer to section **??**.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 (8)	Overall model test statistic for the estimated PVT parameters divided by the test threshold
Padding	u1[..]			Padding bytes, see 4.1.5

(8) 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of satellites for which corrections are provided in this SBF block, i.e. number of SatCorr sub-blocks. If N 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 4.1.5

SatCorr sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
SVID	u1			Satellite ID, see 4.1.9
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 4.1.5

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 **lstAntennaInfo, overview**. (see the description of the commands **setAntennaOffset** and **lstAntennaInfo** 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 section 2.1.4 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of baselines for which relative position, velocity and direction are provided in this SBF block, i.e. number of VectorInfoCart sub-blocks. If N is 0, there are no baseline available for this epoch.
SBLength	u1	1 byte		Length of one sub-block
VectorInfoCart	...	...		*A succession of N VectorInfoCart sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

`VectorInfoCart` sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
`nrSV`	u1			Number of satellites for which corrections are available from the base station identified by the `ReferenceID` field.
`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.
`Mode`	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).
`Misc`	u1			Bit field containing miscellaneous flags: Bit 0: Set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown. 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. Bit 2: Proprietary. Bits 3-7: Reserved
`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 ReferenceID. If bit *i* is set, corrections for the signal type having index *i* are available. The signal numbers are listed in section 4.1.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.
Padding	u1[..]			Padding bytes, see 4.1.5

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 section 2.1.4 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of baselines for which relative position, velocity and direction are provided in this SBF block, i.e. number of VectorInfoGeod sub-blocks. If N is 0, there are no baseline available for this epoch.
SBLength	u1	1 byte		Length of one sub-block
VectorInfoGeod	...	...		*A succession of N VectorInfoGeod sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

VectorInfoGeod sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
NrSV	u1			Number of satellites for which corrections are available from the base station identified by the ReferenceID field.
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.
Mode	u1			Bit field indicating the PVT mode, as follows: Bits 0-3: type of PVT solution: 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) Bits 4-5: Reserved 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. Bit 7: 2D/3D flag: set in 2D mode (height assumed constant and not computed).
Misc	u1			Bit field containing miscellaneous flags: Bit 0: Set if the baseline points to the base station ARP. Unset if it points to the antenna phase center, or if unknown. 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. Bit 2: Proprietary. Bits 3-7: Reserved
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 4.1.10. Bit 0 (GPS-C/A) is the LSB of SignalInfo.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.9 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 **setGNSSAttitude**). 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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 **setGNSSAttitude**). 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	$-2 \cdot 10^{10}$	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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.10 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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			Bit field indicating the synchronization level of the receiver time. If bits 0 to 2 are set, full synchronization is achieved: Bit 0: WNSET: if this bit is set, the receiver week number is set. Bit 1: TOWSET: if this bit is set, the receiver time-of-week is set to within 20ms. Bit 2: FINETIME: if this bit is set, the receiver time-of-week is within the limit specified by the **setClockSyncThreshold** command. Bits 3-7: Reserved
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.11 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
Mode	u1			0: RTCMv2 1: CMRv2 2: RTCMv3 3: RTCMV (a proprietary variant of RTCM2)
Source	u1			Indicates the receiver connection from which the message has been received: 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*]			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: `N = 2 + ((RTCM2Words[1]»9) & 0x1f);` *N* can range from 2 to 33. The first two words are the RTCM2 message header and they are always present. Each of the words is organized as follows: 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. Bits 6-29: 24 information-containing bits of the word. The first received bit is the MSB. Bits 30-31: bit 0 and 1 of the preceding word
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 4.1.5

`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	Do-Not-Use	Description
`Sync1`	c1			Block Header, see 4.1.1
`Sync2`	c1			
`CRC`	u2			
`ID`	u2			
`Length`	u2	1 byte		
`TOW`	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
`WNc`	u2	1 week	65535	
`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 `Datum` field
`Y`	f8	1 m		Antenna Y coordinate
`Z`	f8	1 m		Antenna Z coordinate
`Padding`	u1[..]			Padding bytes, see 4.1.5

RTCMDatum	Number:	4049
	"OnChange" interval:	block generated each time a set of transformation parameters is received

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 Datum field in the PosLocal and PosProjected SBF blocks. Datum is set to 255 if this SourceCRS/TargetCRS 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 PosLocal and the PosProjected SBF blocks: 0: Geometrical height 1: Physical height (height definition in target CRS) 2: Physical height (height definition in source CRS)
QualityInd	u1			Bit field indicating the maximum approximation error after applying the transformation: Bits 0-3: horizontal quality indicator: 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) Bits 4-7: vertical quality indicator, same definition as bits 0-3.
Padding	u1[..]			Padding bytes, see 4.1.5

4.2.12 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of channels for which status are provided in this SBF block, i.e. number of ChannelSatInfo sub-blocks. If N is 0, there are no active channels available for this epoch.
SB1Length	u1	1 byte		Length of a ChannelSatInfo sub-block, excluding the nested ChannelStateInfo sub-blocks
SB2Length	u1	1 byte		Length of a ChannelStateInfo sub-block
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
SatInfo	...	...		*A succession of N ChannelSatInfo sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

ChannelSatInfo sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
SVID	u1			Satellite ID, see 4.1.9
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			bit field:
		1 degree	511	Bits 0-8: Azimuth [0,359]. 0 is North, and Azimuth increases towards East.
				Bits 9-13: Reserved
			3	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 4.1.11.
Reserved2	u1			Reserved for future use, to be ignored by decoding software
Padding	u1[..]			Padding bytes, see 4.1.5
StateInfo	...	...		*A succession of N2 ChannelStateInfo sub-blocks, see definition below*

ChannelStateInfo sub-block definition:

Parameter	Type	Units	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 4.1.5

ReceiverStatus	Number: 4014
	"OnChange" interval: 1s

The ReceiverStatus block provides general information on the status of the receiver.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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			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. 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. 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. 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. 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. Bits 4-7: Reserved
UpTime	u4	1 s		Number of seconds elapsed since the start-up of the receiver, or since the last reset.

RxState	u4			Bit field indicating the status of key components of the receiver: Bit 0: Reserved Bit 1: Reserved Bit 2: EXT_REF: this bit is set if an external frequency reference is detected at the 10 MHz input, and cleared if the receiver uses its own internal clock. 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. Bit 4: WNSET: see corresponding bit in the SyncLevel field of the ReceiverTime block. Bit 5: TOWSET: see corresponding bit in the SyncLevel field of the ReceiverTime block. Bit 6: FINETIME: see corresponding bit in the SyncLevel field of the ReceiverTime block. 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. 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. Bit 9: DISK_MOUNTED: this bit is set when the internal disk (DSK1) is mounted. Bit 10: INT_ANT: this bit is set when the GNSS 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). 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. Bit 12: LBAND_ANT: this bit is set when the L-band signal is tracked from the dedicated L-band antenna, and cleared when it is tracked from the same antenna as the GNSS signals, or when the receiver does not support L-band tracking. Bits 13-31: Reserved

RxError	u4			Bit field indicating whether an error occurred previously. If this field is not equal to zero, at least one error has been detected. Bit 0: Reserved Bit 1: Reserved Bit 2: Reserved Bit 3: SOFTWARE: set upon detection of a software warning or error. This bit is reset by the command "**lif, error**". Bit 4: WATCHDOG: set when the watchdog expired at least once since the last power-on. Bit 5: Reserved 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. Bit 7: Reserved 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. Bit 9: CPUOVERLOAD: set when the CPU load is larger than 90%. Bit 10: INVALIDCONFIG: set if one or more configuration file (e.g. permissions) is invalid or absent. Bit 11: OUTOFGEOFENCE: set if the receiver is currently out of its permitted region of operation (geofencing). Bit 12: Reserved Bit 13: Reserved Bit 14: Reserved Bit 15: Reserved Bit 16: Reserved Bits 17-31: Reserved
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 4.1.5

Rev 1

`AGCState` sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
FrontEndID	u1			Bit field indicating the frontend code and antenna ID: Bits 0-4: frontend code: 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 Bits 5-7: antenna ID: 0 for main, 1 for *Aux1* and 2 for *Aux2*
Gain	i1	1 dB	−128	AGC gain, in dB. The Do-Not-Use value is used to indicate that the frontend PLL is not locked.
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of satellites for which information is provided in this SBF block, i.e. number of SatInfo sub-blocks.
SBLength	u1	1 byte		Length of one SatInfo sub-block
SatInfo	...	...		*A succession of N SatInfo sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

SatInfo sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
SVID	u1			Satellite ID, see 4.1.9
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of connection descriptors for which communication link statistics are included
SBLength	u1	1 byte		Length of one InputStatsSub sub-block.
InputStats	...	...		*A succession of N InputStatsSub sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

InputStatsSub sub-block definition:

<table>
<tr><th>Parameter</th><th>Type</th><th>Units</th><th>Do-Not-Use</th><th>Description</th></tr>
<tr><td>CD</td><td>u1</td><td></td><td></td><td>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></td></tr>
<tr><td>Type</td><td>u1</td><td></td><td></td><td>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</td></tr>
<tr><td>AgeOfLastMessage</td><td>u2</td><td>1 s</td><td>65535</td><td>Age of the last accepted message.

If the age is older than 65534s, it is clipped to 65534s.</td></tr>
<tr><td>NrBytesReceived</td><td>u4</td><td>1 byte</td><td></td><td>Total number of bytes received [9]</td></tr>
<tr><td>NrBytesAccepted</td><td>u4</td><td>1 byte</td><td>4294967295</td><td>Total number of bytes [9] in messages that passed the check for this type of input (CRC, parity check, ...).

The ratio of NrBytesAccepted to NrBytesReceived gives an indication of the quality of the communication link.</td></tr>
<tr><td>NrMsgReceived</td><td>u4</td><td>1 message</td><td>4294967295</td><td>Total number of messages of type Type received.</td></tr>
<tr><td>NrMsgAccepted</td><td>u4</td><td>1 message</td><td>4294967295</td><td>Total number of messages of type Type that were interpreted and used by the receiver.

The ratio of NrMsgAccepted to NrMsgReceived gives an indication of the bandwidth usage efficiency</td></tr>
<tr><td>Padding</td><td>u1[..]</td><td></td><td></td><td>Padding bytes, see 4.1.5</td></tr>
</table>

[9] Note that, for RTCM 2.x, one 8-bit byte contains 6 RTCM data bits.

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N1	u1			Number of OutputStatsSub sub-blocks in this OutputLink block.
SB1Length	u1	1 byte		Length of an OutputStatsSub sub-block, excluding the nested OutputTypeSub sub-block
SB2Length	u1	1 byte		Length of an OutputTypeSub sub-block
Reserved	u1[3]			Reserved for future use
OutputStats	...	...		*A succession of N1 OutputStatsSub sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

`OutputStatsSub` sub-block definition:

<table>
<tr><th>Parameter</th><th>Type</th><th>Units</th><th>Description</th></tr>
<tr><td>CD</td><td>u1</td><td></td><td>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></td></tr>
<tr><td>N2</td><td>u1</td><td></td><td>Number of OutputTypeSub sub-blocks included at the end of this OutputStatsSub sub-block</td></tr>
<tr><td>AllowedRate</td><td>u2</td><td>1 kbyte / s</td><td>Maximum datarate recommended on this connection</td></tr>
<tr><td>NrBytesProduced</td><td>u4</td><td>1 byte</td><td>Total number of bytes produced by the receiver</td></tr>
<tr><td>NrBytesSent</td><td>u4</td><td>1 byte</td><td>Total number of bytes actually sent (i.e. without congestions or transmission errors).

The ratio of NrBytesSent to NrBytesProduced gives an indication of the amount of bandwidth overload.</td></tr>
<tr><td>NrClients</td><td>u1</td><td></td><td>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 NrClients is more than one, the fields NrBytesProduced and NrBytesSent are the total number of bytes produced and sent to all client.</td></tr>
<tr><td>Reserved</td><td>u1[3]</td><td></td><td>Reserved for future use</td></tr>
<tr><td>Padding</td><td>u1[..]</td><td></td><td>Padding bytes, see 4.1.5</td></tr>
<tr><td>OutputType</td><td>...</td><td>...</td><td>A succession of N2 OutputTypeSub sub-blocks, see definition below</td></tr>
</table>

Rev 1

`OutputTypeSub` sub-block definition:

Parameter	Type	Units	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 4.1.5

NTRIPClientStatus	Number: 4053 "OnChange" interval: 1s

This block reports the current status of the NTRIP client connections.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of NTRIP client connections for which status is provided in this block, i.e. number of NTRIPClientConnection sub-blocks.
SBLength	u1	1 byte		Length of one NTRIPClientConnection sub-block
NTRIPClientConnection	...	...		*A succession of N NTRIPClientConnection sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

NTRIPClientConnection sub-block definition:

Parameter	Type	Units	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 InputLink block. 3: Error detected, the error code is provided in the next field. 4: Retrying, client encountered an error, we are trying to reconnect. 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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 4.1.5

WiFiAPStatus	Number: 4054
	"OnChange" interval: 1s

This block contains the hostname and the MAC and IP addresses of the receiver when configured in WiFi access point. It also contains the list of all connected clients.

The current WiFi mode is reported in the Mode argument. When the receiver is configured in WiFi client mode or when WiFi is disabled, many fields are not applicable and are set to their do-not-use value.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of WiFi clients currently connected to the receiver.
SBLength	u1	1 byte		Length of one WiFiClient sub-block
APIPAddress	u1[16]		All elements set to 0	IP address of the WiFi access point. 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.
Mode	u1			WiFi mode: 0: WiFi disabled 1: WiFi enabled in access point mode 2: WiFi enabled in client mode
Hotspot	u1			WiFi hotspot: 0: Hotspot disabled 1: Hotspot enabled and no Internet access 2: Hotspot enabled and Internet access
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software
WiFiClient	...	...		*A succession of N WiFiClient sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

WiFiClient sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
ClientHostName	c1[32]			Hostname of a WiFi client connected to the receiver, or empty if not known.
ClientMACAddress	u1[6]			MAC address of a WiFi client connected to the receiver. The first byte corresponds to the MSB of the address.
ClientIPAddress	u1[16]		All elements set to 0	IP address of a WiFi client connected to the receiver. 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.
Padding	u1[..]			Padding bytes, see 4.1.5

WiFiClientStatus	Number: 4096 "OnChange" interval: 1s

This block contains WiFi status information of the receiver when configured in WiFi client mode.

When the receiver is not configured in WiFi client mode, many fields are not applicable and are set to their do-not-use value.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
SSID_AP	c1[32]			SSID of the access point the receiver is currently connected to. Empty when not connected.
IPAddress	u1[16]		All elements set to 0	IP address of the receiver as WiFi client. 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. All bytes are set to zero if the IP address is not applicable or not known yet (e.g. when the receiver is currently obtaining its IP address from the access point).
Reserved	u1[1]			Reserved for future use, to be ignored by decoding software
SigLevel	i1	1 dBm	−128	WiFi signal power level
Status	u1			Bit field: Bits 0-3: WiFi client connection status: 0: Not connected, see the ErrorCode field for reason 1: Connecting 2: Connected Bits 4-7: Reserved
ErrorCode	u1			WiFi client error code: 0: No error 1: WiFi disabled or not in client mode 2: No reachable WiFi access point found 3: No known access point in reach (use the **exeAddWiFiAccessPoint** command to add an access point to the list of known access points) 4: Known access points were found but the receiver could not connect to them
Padding	u1[..]			Padding bytes, see 4.1.5

CellularStatus	Number: 4055
	"OnChange" interval: 1s

This block contains the cellular status information.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
ConnectionType	u1		255	Type of current connection: 0: Not connected 1: 2G: GPRS (up to 114 kbit/s) 2: 2G: EDGE (up to 237 kbit/s) 3: 3G: UMTS (up to 384 kbit/s) 4: 3G: HSDPA (up to 7.2 Mbit/s) 5: 3G: HSUPA (up to 5.4 Mbit/s) 6: 3G: HSPA (up to 7.2 Mbit/s)
RSSI	i1	1 dBm	127	Cellular Received Signal Strength Indicator
OperatorName	c1[20]			Name of the cellular operator, right-padded with zeros.
Status	u1			Cellular status: 0: Cellular disabled 1: Initializing 2: Cellular in standby mode 3: Connecting 4: Connected 5: Disconnecting 6: Searching 254: Error detected, the error code is provided in the next field.
ErrorCode	u1			Cellular error code: 0: No error 1: SIM card not detected 2: PIN code invalid 3: Connection error 4: SIM card blocked, PUK code needed (use **exeUnblockCellular** command) 5: Modem not responding 255: Unknown error
Padding	u1[..]			Padding bytes, see 4.1.5

BluetoothStatus	Number: 4051 "OnChange" interval: 1s

This block contains the list of Bluetooth devices currently paired to the receiver.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of paired Bluetooth devices reported in this block.
SBLength	u1	1 byte		Length of one BTDevice sub-block
Mode	u1			Bit field: Bit 0: Bit set when Bluetooth is enabled with the command **setBTParameters**. Bit 1: Bit set when the receiver is discoverable by other Bluetooth devices (see **setBTParameters**). Bits 2-7: Reserved
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
BTDevice	...	...		*A succession of N BTDevice sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

BTDevice sub-block definition:

Parameter	Type	Units	Description
DeviceName	c1[30]		Paired device name.
Flags	u1		Bit field: Bit 0: Bit set when this Bluetooth device is currently connected to the receiver. Bits 1-7: Reserved
Padding	u1[..]		Padding bytes, see 4.1.5

BatteryStatus	Number: 4083
	"OnChange" interval: 1s

This block reports the status of the N batteries in the receiver.

If bit 2 of the Status field (Battery in use) is unset for all batteries, this means that the receiver is powered through the external power connector.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of batteries for which status is provided in this block, i.e. number of Battery sub-blocks.
SBLength	u1	1 byte		Length of one Battery sub-block.
ExtSupply	u1			Bit field: Bit 0: bit set if an external power supply is present Bit 1: bit set when the external power supply is powerful enough to power the receiver and charge the batteries. If not set, the external power supply will typically only partially power the receiver: the total battery charge level may decrease, eventually causing the receiver to shut down. Bits 2-7: Reserved
Reserved	u1[3]			Reserved for future use, to be ignored by decoding software
Battery	...	...		*A succession of N Battery sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

Battery sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
ChargeLevel	u1	1 %	255	Battery charge level: remaining charge level when not charging, and current charge level when charging.
Status	u1			Bit field: Bit 0: bit set if battery is present Bit 1: bit set when battery is charging Bit 2: bit set when this battery is currently used to power the receiver Bits 3-7: Reserved
RemainingTime	u2	1 minute	65535	Estimated remaining operation time for this battery (set to Do-Not-Use value when in charge, or unknown).
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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]			N successive quality indicators, coded as follows: Bits 0-7: Quality indicator type: 0: Overall quality 1: GNSS signals from main antenna 2: GNSS signals from aux1 antenna 11: RF power level from the main antenna 12: RF power level from the aux1 antenna 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. Bits 8-11: Value of this quality indicator (from 0 for low quality to 10 for high quality) Bits 12-15: Reserved for future use, to be ignored by decoding software.
Padding	u1[..]			Padding bytes, see 4.1.5

DiskStatus	Number: 4059
	"OnChange" interval: 1s

This block reports the size and usage of the disks mounted on the receiver.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of DiskData sub-blocks this block contains.
SBLength	u1	1 byte		Length of one DiskData sub-blocks in bytes.
Reserved	u1[4]			Reserved for future use
DiskData	...	...		*A succession of N DiskData sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

DiskData sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
DiskID	u1			ID of the disk, starting at 1 for the internal SD Memory Card.
Status	u1			Bit field: Bit 0: DISK_MOUNTED: bit set when the disk is mounted. 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. 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. Bit 3: LOGGING_ENABLED: bit set when at least one file is open on the disk, regardless of the logging rate. Bits 4-7: Reserved
DiskUsageMSB	u2		65535 (10)	16 MSB of the total disk usage. The disk usage in bytes is given by DiskUsageMSB*4294967296+DiskUsageLSB.
DiskUsageLSB	u4		4294967295 (10)	32 LSB of the total disk usage. The disk usage in bytes is given by DiskUsageMSB*4294967296+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 4.1.5

(10) The disk usage is invalid if both DiskUsageMSB is 65535 and DiskUsageLSB is 4294967295.

4.2.13 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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
Rev 1 MarkerType	c1[20]			Marker type, this is a 20-character string, right padded with zeros
Rev 2 GNSSFWVersion	c1[40]			Version tag of the GNSS firmware installed on the receiver. This is a 40-character string, right padded with zeros.
Rev 3 ProductName	c1[40]			Product name. This is a 40-character string, right padded with zeros.
Padding	u1[..]			Padding bytes, see 4.1.5

RxComponents	Number: 4084 "OnChange" interval: 10 s

This block contains information on various hardware and software components of the receiver.

Parameter	Type	Units	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
N	u1			Number of components for which information is provided in this block.
SBLength	u1	1 byte		Length of one Component sub-block
Reserved	u1[4]			Reserved for future use, to be ignored by decoding software
Component	...	...		*A succession of N Component sub-blocks, see definition below*
Padding	u1[..]			Padding bytes, see 4.1.5

Component sub-block definition:

Parameter	Type	Units	Do-Not-Use	Description
Type	u1			Type of component described in this sub-block: 1: Motherboard 2: GNSS module 3: WiFi module 4: Cellular module 5: Bluetooth module 6: L-Band module 7: UHF module
CPULoad	u1	1 %	255	Load on the component CPU, if applicable.
Reserved	u1[2]			Reserved for future use, to be ignored by decoding software
Name	c1[40]			Component name.
SerialNumber	c1[20]			Component serial number. Empty if not applicable.
FWVersion	c1[40]			Component firmware version. Empty if not applicable.
MACAddress	u1[6]			MAC address if applicable. The first byte corresponds to the MSB of the address. All bytes set to 0 for components for which no MAC address applies.
Padding	u1[..]			Padding bytes, see 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 4.1.5

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 4.1.5

ASCIIIn	Number: 4075
	"OnChange" interval: block generated each time an ASCII string is received

The ASCIIIn 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 ASCIIIn 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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
CD	u1			Identifier of the connection to which these statistics apply: (see table below)
Reserved1	u1[3]			Reserved, contents to be ignored.
StringLn	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 4.1.5

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	

4.2.14 Deprecated or Obsolete Bocks

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	Do-Not-Use	Description
Sync1	c1			Block Header, see 4.1.1
Sync2	c1			
CRC	u2			
ID	u2			
Length	u2	1 byte		
TOW	u4	0.001 s	4294967295	Receiver time stamp, see 4.1.3
WNc	u2	1 week	65535	
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 4.1.5

4.3 SBF Change Log

Date	Change Description
Feb 04, 2015	Added the QZSNav block containing decoded QZSS navigation data
Jan 13, 2015	Added the PosProjected block containing plane grid coordinates
Dec 12, 2014	Added the base measurements quality indicator
April 30, 2014	Added new values for the Datum field
April 22, 2014	Added the DiskStatus block reporting the disk usage and free space of the disks available on the receiver
Feb 21, 2014	Added the NTRIPClientStatus block for the NTRIP client connection status
June 24, 2013	Added the RxComponents block containing information on the various receiver's components
June 24, 2013	Added the BatteryStatus block
June 24, 2013	Added the BluetoothStatus block for the Bluetooth status
June 24, 2013	Added the WiFiAPStatus block for the WiFi status in access point mode
June 24, 2013	Added the CellularStatus block for the cellular modem status
March 14, 2013	Added the QualityInd block containing various quality indicators
Feb 8, 2013	Fixed typo: field t_oG of GALGstGps changed to type u4 and units of seconds
Jan 8, 2013	Added fields HAccuracy, VAccuracy and Misc to the PVTCartesian and PVTGeodetic blocks
Dec 19, 2012	Added PRNs 139 and 140 to the list of SBAS satellites
Oct 25, 2012	Added RTCMDatum and PosLocal blocks
Oct 1, 2012	Added new signal type for L-band and SBAS L5 signals (value 23 and 25)
Sep 20, 2012	Added field PPPInfo to the PVTCartesian and PVTGeodetic blocks
Feb 28, 2012	Added GALSARRLM block
Feb 6, 2012	Added QZSS signals and QZSRawL1CA, QZSRawL2C and QZSRawL5 blocks

Appendix A

Attitude Angles

The attitude of the vehicle is defined as the angles between the vehicle reference frame and the local-level reference frame (defined by the East, North and Up directions). The vehicle reference frame is defined as follows. It is attached to the vehicle and 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.

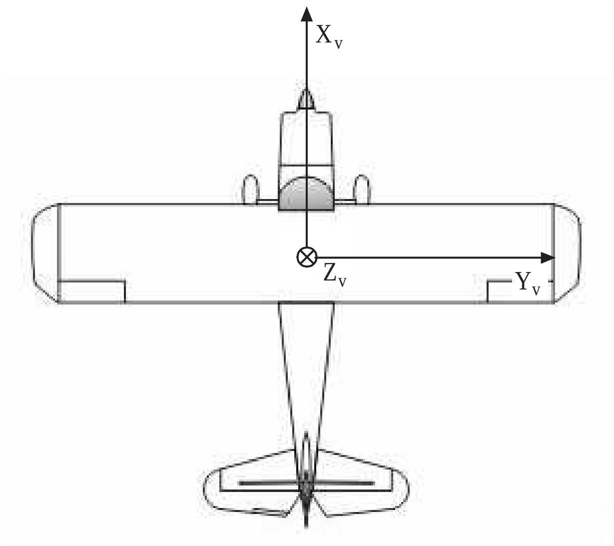

Figure A-1: Vehicle reference frame.

Septentrio receivers express the vehicle attitude in Euler angles using the heading-pitch-roll rotation sequence. More specifically, 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. The heading (ψ) of the vehicle is defined as the right-handed rotation of the vehicle about the Z axis ($0^o \leq \psi \leq 360^o$). The pitch (θ) of the vehicle is defined as the right-handed rotation about the vehicle Y axis ($-90^o \leq \theta \leq 90^o$). The roll ($\phi$) of the vehicle is defined as the right-handed rotation about the vehicle X axis ($-180^o \leq \phi \leq 180^o$).

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;

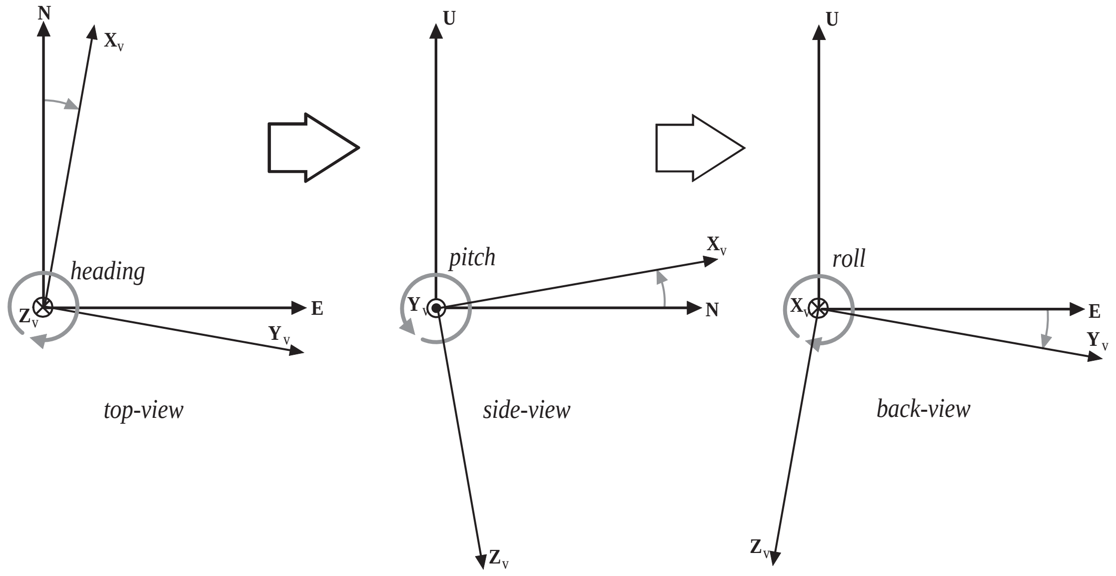

Figure A-2: Euler angle sequence.

Appendix B

List of SBF Blocks

The following table provides the list of the SBF blocks names and numbers available on Altus NR2 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 4.1.1).

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 4.1.8). The "Time stamp" column indicates which type of time is encoded in the block time stamp (see section 4.1.3 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
EndOfMeas	5922	measurement epoch marker	•	•	R
Navigation Page Blocks					
GPSRawCA	4017	GPS CA navigation subframe			S
GPSRawL2C	4018	GPS L2C navigation frame			S
GLORawCA	4026	GLONASS CA navigation string			S
GALRawINAV	4023	Galileo I/NAV navigation page			S
GEORawL1	4020	SBAS L1 navigation message			S
QZSRawL1CA	4066	QZSS L1 CA navigation frame			S
QZSRawL2C	4067	QZSS L2C 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

Block name	Block No	Content description	Flex Rate	esoc	Time Stamp
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
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

Block name	Block No	Content description	Flex Rate	esoc	Time Stamp
Receiver Time Blocks					
`ReceiverTime`	5914	Current receiver and UTC time	•	•	R
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
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
`WiFiAPStatus`	4054	WiFi status in access point mode		•	R
`WiFiClientStatus`	4096	WiFi status in client mode		•	R
`CellularStatus`	4055	Cellular status		•	R
`BluetoothStatus`	4051	Bluetooth status		•	R
`BatteryStatus`	4083	Battery status		•	R
`QualityInd`	4082	Quality indicators		•	R
`DiskStatus`	4059	Internal logging status		•	R
Miscellaneous Blocks					
`ReceiverSetup`	5902	General information about the receiver set-up		•	R
`RxComponents`	4084	Information on various receiver components		•	R
`Commands`	4015	Commands entered by the user		•	R
`Comment`	5936	Comment entered by the user		•	R
`ASCIIIn`	4075	Search-and-rescue return link message			R
Deprecated or Obsolete Bocks					
`BaseLine`	5950				R

Appendix C

List of NMEA Sentences

The following table provides a list of the NMEA messages supported by your receiver. The first column is the message identifier to be used in the **setNMEAOutput** and the **exeNMEAOnce** commands.

For a full description of the NMEA messages, please refer to the NMEA 0183 standard.

Message Identifier	NMEA Formatter	Short Description	Comment
GGA	GGA	GPS Fix Data	GPS Quality Indicator field is set to 5 in PPP mode
GGQ	GGQ	Leica Real-Time Position with CQ	
GLL	GLL	Geographic Position - Latitude/Longitude	
GRS	GRS	GNSS Range Residuals	
GSA	GSA	GNSS DOP and Active Satellites	
GST	GST	GNSS Pseudorange Error Statistics	
GSV	GSV	GNSS Satellites in View	
LLK	LLK	Leica Local Position and GDOP	
LLQ	LLQ	Leica Local Position and Quality	
RMC	RMC	Recommended Minimum Specific GNSS Data	
SBT	SBT	Battery Status	Septentrio proprietary, see section C.1.1
SCL	SCL	Cellular Status	Septentrio proprietary, see section C.1.2
SNC	SNC	NTRIP Client Status	Septentrio proprietary, see section C.1.3
TFM	TFM	Used Coordinate Transformation Messages	Septentrio proprietary, see section C.1.4
TXTbase	TXT	Text Transmission	Text from a base station in RTCM message type 1029. 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	VTG	Course Over Ground and Ground Speed	
ZDA	ZDA	Time and Date	

Appendix C.1 Proprietary NMEA Sentences

C.1.1 SBT: Battery Status

This proprietary sentence is the NMEA equivalent of the `BatteryStatus` SBF block.

Field	Description
$PSSN,SBT,	Start of sentence
[	
x,	message revision
xxxxxxxxx,	time of week, milliseconds
xxxx,	week number
x.x,	`ExtSupply` field of the `BatteryStatus` SBF block
,	Reserved
`<SBTSub>`	a succession of `SBTSub` sub-messages, see definition below
]	
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

`SBTSub` definition:

Field	Description
[	
x.x,	`ChargeLevel` field of the `BatteryStatus` SBF block
x.x,	`Status` field of the `BatteryStatus` SBF block
x.x	`RemainingTime` field of the `BatteryStatus` SBF block
]	

Example:
`$PSSN,SBT,[0,474898000,1840,0,,[91,5,173],[100,1,183]]*01`

C.1.2 SCL : Cellular Status

This proprietary sentence is the NMEA equivalent of the `CellularStatus` SBF block.

Field	Description
$PSSN,SCL,	Start of sentence
[	
x,	message revision
xxxxxxxxx,	time of week, milliseconds
xxxx,	week number
x.x,	`ConnectionType` field of the `CellularStatus` SBF block
x.x,	`RSSI` field of the `CellularStatus` SBF block
c–c,	`OperatorName` field of the `CellularStatus` SBF block
x.x,	`Status` field of the `CellularStatus` SBF block
x.x	`ErrorCode` field of the `CellularStatus` SBF block
]	
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

Example:
`$PSSN,SCL,[0,379359000,1840,6,-83,"BEL PROXIMUS",4,0]*52`

C.1.3 SNC : NTRIP Client Status

This proprietary sentence is the NMEA equivalent of the `NTRIPClientStatus` SBF block.

Field	Description
$PSSN,SNC,	Start of sentence
[	
x,	message revision
xxxxxxxxx,	time of week, milliseconds
xxxx	week number
`<SNCSub>`	a succession of `SNCSub` sub-messages, see definition below
]	
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

`SNCSub` definition:

Field	Description
[	
x.x,	`CDIndex` field of the `NTRIPClientStatus` SBF block
x,	`Status` field of the `NTRIPClientStatus` SBF block
x.x	`ErrorCode` field of the `NTRIPClientStatus` SBF block
]	

Example:
`$PSSN,SNC,[0,379359000,1840,[1,2,0]]*0D`

C.1.4 TFM : Used RTCM Coordinate Transformation Messages

This proprietary sentence indicates which RTCM coordinate transformation messages have been received and used in the position computation.

Field	Description
$PSSN,TFM,	Start of sentence
hhmmss.ss,	UTC time (HoursMinutesSeconds.DecimalSeconds)
x,	Height indicator, a copy of the Height Indicator field in RTCM message 1021 or 1022. Null if unknown.
xxxx,	Message 1021/1022 usage (they are exclusive). Possible field values: 1021: Message type 1021 used; 1022: Message type 1022 used; null: neither 1021 nor 1022 used.
xxxx,	Message 1023/1024 usage (they are exclusive). Possible field values: 1023: Message type 1023 used; 1024: Message type 1024 used; null: neither 1023 nor 1024 used.
xxxx	Message 1025/1026/1027 usage (they are exclusive). Possible field values: 1025: Message type 1025 used; 1026: Message type 1026 used; 1027: Message type 1027 used; null: neither 1025 nor 1026 nor 1027 used.
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

Example:

```
$PSSN,TFM,104751.00,2,1021,1023,1025*5F
```

Appendix D

List of CMR and RTCM Messages

This appendix provides a list of all the CMR and RTCM (v2.x and v3.x) messages supported by the receiver.

Appendix D.1 CMR Messages

Message Identifier	Short Description
CMR0	Observables
CMR1	Reference Station Coordinates
CMR2	Reference Station Description
CMR3	GLONASS Observables

Appendix D.2 RTCM v2.x Messages

Message Identifier	Short Description
RTCM1	Differential GPS Corrections
RTCM3	GPS Reference Station Parameters
RTCM9	GPS Partial Correction Set
RTCM16	GPS Special Message
RTCM17	GPS Ephemerides Message
RTCM18	RTK Uncorrected Carrier Phases
RTCM19	RTK Uncorrected Pseudoranges
RTCM20	RTK Carrier Phase Corrections
RTCM21	RTK/Hi-Accuracy Pseudorange Corrections
RTCM22	Extended Reference Station Parameters
RTCM23	Antenne Type Definition Record
RTCM24	Antenna Reference Point (ARP)
RTCM31	Differential GLONASS Corrections
RTCM32	GLONASS Reference Station Parameters

Appendix D.3 RTCM v3.x Messages

Message Identifier	Short 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
RTCM1019	GPS Satellite Ephemeris Data
RTCM1020	Glonass 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 Pseudoranges
RTCM1072	GPS MSM2, Compact PhaseRanges
RTCM1073	GPS MSM3, Compact Pseudoranges and PhaseRanges
RTCM1074	GPS MSM4, Full Pseudoranges and PhaseRanges plus CNR
RTCM1075	GPS MSM5, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1076	GPS MSM6, Full Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1077	GPS MSM7, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1081	GLONASS MSM1, Compact Pseudoranges
RTCM1082	GLONASS MSM2, Compact PhaseRanges
RTCM1083	GLONASS MSM3, Compact Pseudoranges and PhaseRanges
RTCM1084	GLONASS MSM4, Full Pseudoranges and PhaseRanges plus CNR
RTCM1085	GLONASS MSM5, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1086	GLONASS MSM6, Full Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1087	GLONASS MSM7, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1091	GALILEO MSM1, Compact Pseudoranges
RTCM1092	GALILEO MSM2, Compact PhaseRanges
RTCM1093	GALILEO MSM3, Compact Pseudoranges and PhaseRanges
RTCM1094	GALILEO MSM4, Full Pseudoranges and PhaseRanges plus CNR
RTCM1095	GALILEO MSM5, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1096	GALILEO MSM6, Full Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1097	GALILEO MSM7, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)
RTCM1111	QZSS MSM1, Compact Pseudoranges
RTCM1112	QZSS MSM2, Compact PhaseRanges
RTCM1113	QZSS MSM3, Compact Pseudoranges and PhaseRanges

Message Identifier	Short Description
RTCM1114	QZSS MSM4, Full Pseudoranges and PhaseRanges plus CNR
RTCM1115	QZSS MSM5, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR
RTCM1116	QZSS MSM6, Full Pseudoranges and PhaseRanges plus CNR (high resolution)
RTCM1117	QZSS MSM7, Full Pseudoranges, PhaseRanges, PhaseRangeRate and CNR (high resolution)

Index of Commands

S

U

W

Index of SBF Blocks