Firmware User Manual

Applicable to SSRC5 2.3.2

Firmware User Manual

Revision 0, September 23, 2011

Applicable to SSRC5 2.3.2

Septentrio Satellite Navigation
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

List of Contents

List of Figures

List of Acronyms

APME	A Posteriori Multipath Estimation
ARP	Antenna Reference Point
ASCII	American Standard Code for Information Interchange
CMR	Compact Measurement Record
CPU	Central Processing Unit
CR	Carriage Return
DGPS	Differential Global Positioning System
DHCP	Dynamic Host Configuration Protocol
DOP	Dilution of Precision
EGNOS	European Geostationary Navigation Overlay System
ESTB	EGNOS System Test Bed
FPGA	Field Programmable Gate Array
GIOVE	Galileo In-Orbit Validation Element
GLONASS	Global Orbiting Navigation Satellite System (Russian alternative for GPS)
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPX	GPS eXchange
GUI	Graphical User Interface
HERL	Horizontal External Reliability Level
HPL	Horizontal Protection Level
IGS	International GNSS Service
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KML	Keyhole Markup Language
LAMBDA	Least-squares Ambiguity Decorrelation Adjustment
LED	Light Emitting Diode
MDB	Minimal Detectable Bias
MOPS	Minimum Operational Performance Standards
MT	Message Type
NMEA	National Marine Electronics Association
OTF	On the Fly
PC	Phase Center
PPP	Precise Point Positioning
PPS	Pulse Per Second
PVT	Position Velocity Time
RAIM	Receiver Autonomous Integrity Monitoring
RINEX	Receiver Independent Exchange Format
RTCA	Radio Technical Commission for Aeronautics
RTCM	Radio Technical Commission for Maritime Services

RTK	Real Time Kinematic
RTS	Request to Send
SBAS	Space Based Augmentation System
SBF	Septentrio Binary Format
SD	Secure Digital
SDHC	Secure Digital High Capacity
SIS	Signal In Space
SNMP’	Simple Network Management Protocol (Septentrio variant)
TOW	Time Of Week
USB	Universal Serial Bus
UTC	Coordinated Universal Time
VERL	Vertical External Reliability Level
VPL	Vertical Protection Level
WAAS	Wide Area Augmentation System
WN	Week Number
XERL	External Reliability Levels

1 Quick Start

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

1.1 Quick Start Equipment

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

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

1.2 Quick Start Procedure

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

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

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

Step 4 Start RxControl:

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

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

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

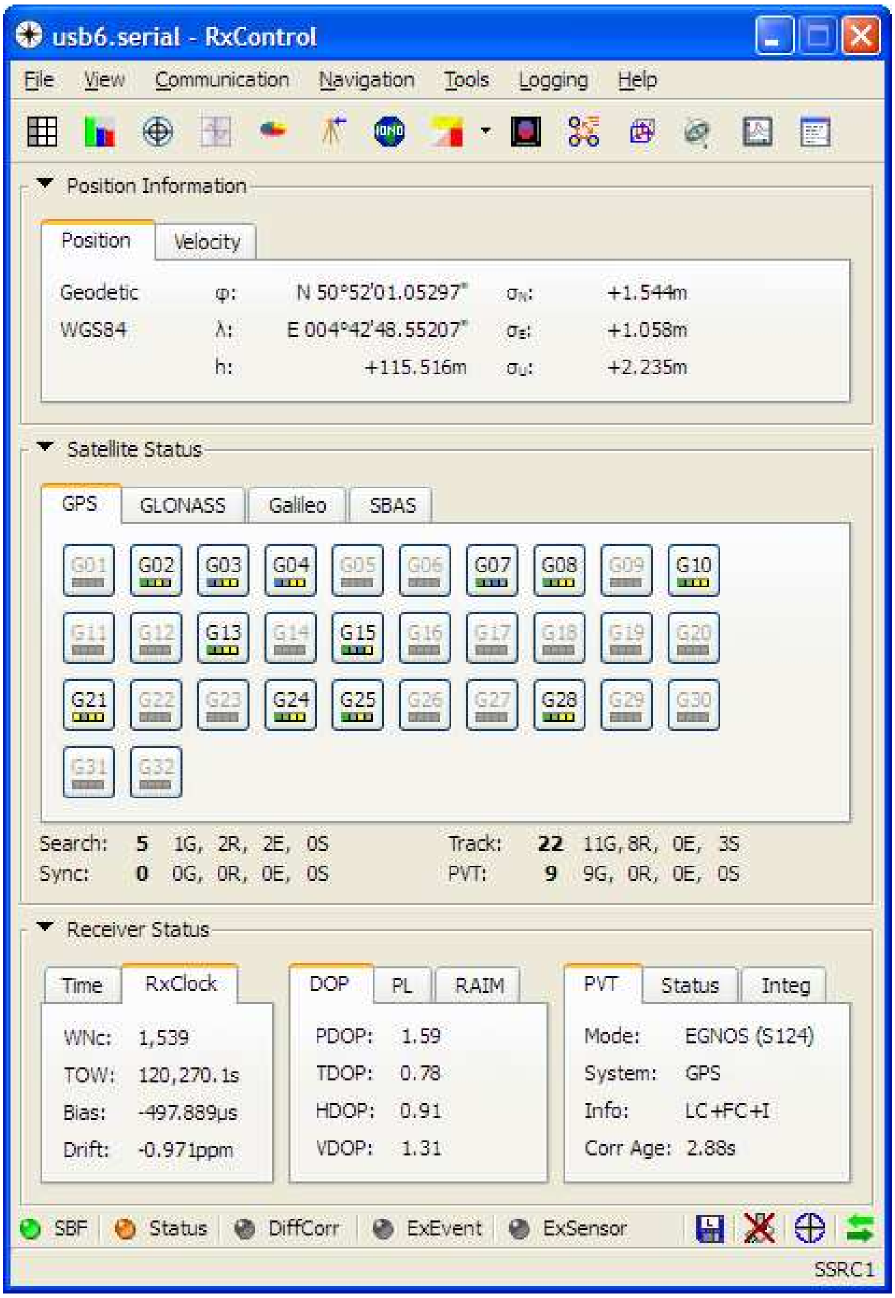

Figure 1-1: RxControl main window.

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

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

2 How To...

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

You can enter user commands in many different ways:

- You can enter commands manually through one of the receiver input ports (see section 2.1). In this chapter, user commands are referred to by their full name for readability. When typing the command, you can always use the short mnemonic equivalent to save typing effort. For instance, instead of typing **`setCOMSettings`**, you can type **`scs`**. See the Command Line Interface Reference Guide to know the mnemonic equivalent of a given command.
- You can type commands or mnemonics in the console window of RxControl (menu *Tools > Expert Console*).
- All commands can also be accessed graphically through menus in RxControl.

⚠ Depending on the capabilities of your particular receiver (see section 2.14), some of the features described here may not be supported.

2.1 Connect to the Receiver

2.1.1 Via COM Ports

The most straightforward 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 operating system you are running on your PC is of no importance; you should only be able to run a terminal emulation program (like HyperTerminal on Windows or minicom on Linux) with full access to the COM port to which the receiver is connected.

To get connected, attach the serial cable, power on the receiver, and launch your terminal program. Make sure that it uses the correct port settings. The default settings are:

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

The baud rate can be modified at any time by using the **`setCOMSettings`** command.
RxControl: *Communication > COM Port Settings*

Since the receiver does not echo the incoming characters, it is handy to enable the local-echo feature of the terminal emulation program in order to see the characters you are typing.

The easiest way to find out whether your physical and logical connection is established is to press the <Enter> key. If the connection is correctly established, the receiver should reply with a prompt.

2.1.2 Via USB

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

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

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

2.1.3 Connection Descriptors

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

COMx: one of the native serial ports;

USBx: one of the virtual serial ports, built on top of the USB interface;

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

2.2 Understand the Output of the Receiver

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

Proprietary messages	First two characters
ASCII command replies and command error notification	`$R`
ASCII transmissions (e.g. periodic output of the status screen), terminated by a prompt. Two sub-types are defined: • `$TD` : ASCII display generated by the receiver; • `$TE` : event notification (e.g. receiver is shutting down).	`$T`
Formatted information blocks (e.g. formal command description)	`$-`
SNMP' binary command replies (Septentrio proprietary)	`$&`
Proprietary binary data (SBF)	`$@`

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

2.2.1 Proprietary Binary Output (SBF)

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

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

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

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

2.2.2 NMEA

The receiver can generate a set of approved NMEA sentences, which conform to the NMEA Standard[1]. The benefit of the NMEA format is that it is standardized. Many electronic devices and software packages support NMEA. The drawback of NMEA is a relatively low level of detail. Appendix B provides a short overview of selected NMEA sentences.

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

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

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

2.2.3 RTCM and CMR

If this feature is enabled in your receiver, the receiver can operate as DGPS and/or RTK base station and output the corresponding RTCM or CMR messages. The instructions to set the receiver in base station mode can be found in section 2.5. Appendix B provides a short overview of supported RTCM and CMR messages.

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

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

2.3 Output SBF

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

1. First make sure that the USB2 connection is configured for SBF output (this is the default). In case this is not so, you should invoke:
   `setDataInOut,USB2, ,+SBF <CR>`
   RxControl: *Communication > Input/Output Selection*

2. Scheduling SBF blocks for output is done by defining so-called "SBF streams". Up to 10 SBF streams can be defined by the user. A stream consists of a set of SBF blocks that need to be output at a given rate on a given connection descriptor. By default, all streams are empty, and no SBF blocks are output. For our example, we will need to use two streams: the first one for the `MeasEpoch` and `PVTCartesian` SBF blocks at a 10-Hz rate, and the second one for the `GPSNav` at the "`OnChange`" rate. Defining these SBF streams involves the **`setSBFOutput`** command:
   `setSBFOutput,Stream1,USB2,MeasEpoch+PVTCartesian,msec100 <CR>`
   `setSBFOutput,Stream2,USB2,GPSNav,OnChange <CR>`
   RxControl: *Communication > Output Settings > SBF Output*

   If you want to output the same SBF blocks at the same rate on another connection, say, COM1, you will need to use two additional streams, for instance `Stream3` and `Stream4`:
   `setSBFOutput,Stream3,COM1,MeasEpoch+PVTCartesian,msec100 <CR>`
   `setSBFOutput,Stream4,COM1,GPSNav,OnChange <CR>`

3. To stop outputting SBF on a given connection, you can either redefine or empty the corresponding streams:
   `setSBFOutput,Stream1,USB2,none <CR>`
   `setSBFOutput,Stream2,USB2,none <CR>`

   A second possibility is to disable all SBF messages on that connection:
   `setDataInOut,USB2, ,-SBF <CR>`
   RxControl: *Communication > Input/Output Selection*

2.4 Save the Configuration in Non-Volatile Memory

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

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

At any time, it is possible to save the current receiver configuration into non-volatile memory, in order to force the receiver to always start up in that configuration. To do so, the following command should be entered:
`exeCopyConfigFile,Current,Boot <CR>`
RxControl: *File > Copy Configuration*

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

2.5 Configure the Receiver in DGPS/RTK-Base Mode

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

2.5.1 Static Base Station Mode

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

1. Connect the receiver to a survey-grade antenna at a fixed location.

2. For accurate and repetitive absolute positioning, you must provide the accurate coordinates of the antenna reference point (ARP). The ARP usually corresponds to the center of the bottom of the antenna (see also section 3.4.3.6). 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>`
   RxControl: *Navigation > Positioning Mode > PVT Mode*

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

3. When the PVT engine operates in static mode, the PVT residuals are generally larger than in rover mode (because only the clock term is estimated). Depending on the selected RAIM thresholds, RAIM may remove too many wrongly identified outliers (see also section 3.5). This behaviour will be more visible if the ARP coordinates are not accurately set. A measurement that has been identified as outlier in the base station will not be included in the RTCM and CMR messages. For best performance, it is recommended to use non-default values for the RAIM probability of false alarm and model reliability. The following settings are recommended:
   `setRAIMLevels,on,-2,-2,-3 <CR>`
   RxControl: *Navigation > Receiver Operation > Position > Integrity*

4. 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 3.4.3.6). For instance:
   `setAntennaOffset,Main, , , ,"AT2775-54SW","5684" <CR>`
   RxControl: *Navigation > Receiver Setup > Antennas*

5. Use the commands **`setRTCMv2Interval`**, **`setRTCMv2IntervalObs`**, **`setRTCMv3Interval`** or **`setCMRv2Interval`** to specify the message interval. The default interval is given in the description of these commands in the Command Line Interface Reference Guide. For instance, to change the default interval at which RTCM 2.x message type 3 is generated to 6 seconds, type:
   `setRTCMv2Interval,RTCM3,10 <CR>`
   RxControl: *Communication > Output Settings > Differential Corrections > RTCMv2*

6. Use the commands **setRTCMv2Formatting**, **setRTCMv3Formatting** or **setCMRv2-Formatting** to specify the base station ID. If you are setting up multiple base stations, make sure to select a unique ID for each of them. For instance:
   setRTCMv2Formatting,496 <CR>
   RxControl: *Communication > Output Settings > Differential Corrections > RTCMv2*

7. Specify the baud rate of the serial port over which the RTCM or CMR messages have to be sent. For instance if the differential correction stream needs to be output on COM2 at 9600 baud, use:
   setCOMSettings,COM2,baud9600 <CR>
   RxControl: *Communication > COM Port Settings*

8. It is recommended to enable code smoothing in order to mitigate propagation of multipath at the base station into the DGPS corrections and RTK data. For instance to smooth all pseudoranges with a smoothing length of 900s, use:
   setSmoothingInterval,all,900 <CR>
   RxControl: *Navigation > Receiver Operation > Tracking and Measurements > Smoothing*

9. According to the RTCM standard, an RTK base station must keep its clock error under 1.1 milliseconds. The CMR standard is even more stringent with a prescribed maximum clock error of 0.5ms (which is the receiver default). In case the receiver is not in its default configuration, you can restore the default setting by using:
   setClockSyncThreshold,usec500 <CR>
   RxControl: *Navigation > Receiver Operation > Timing*

10. By default, the receiver is configured to output all RTCM and CMR messages necessary for DGPS and RTK operation. In case the default has been modified, use the commands **set-RTCMv2Output**, **setRTCMv3Output** or **setCMRv2Output** to specify which types of messages to enable for output. For instance, to output RTCM2.x messages 1 and 3 on COM2, use:
    setRTCMv2Output,COM2,RTCM1+RTCM3 <CR>
    RxControl: *Communication > Output Settings > Differential Corrections > RTCMv2*

11. The connection which needs to output the RTCM stream must be configured to do so. For instance, to enable RTCM 2.x output through COM2, use:
    setDataInOut,COM2,,RTCMv2 <CR>
    RxControl: *Communication > Input/Output Selection*

To stop transmitting RTCM messages, enter the following command:
setDataInOut,COM2,,none <CR>
RxControl: *Communication > Input/Output Selection*

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

2.5.2 RTK Moving Base Station Mode

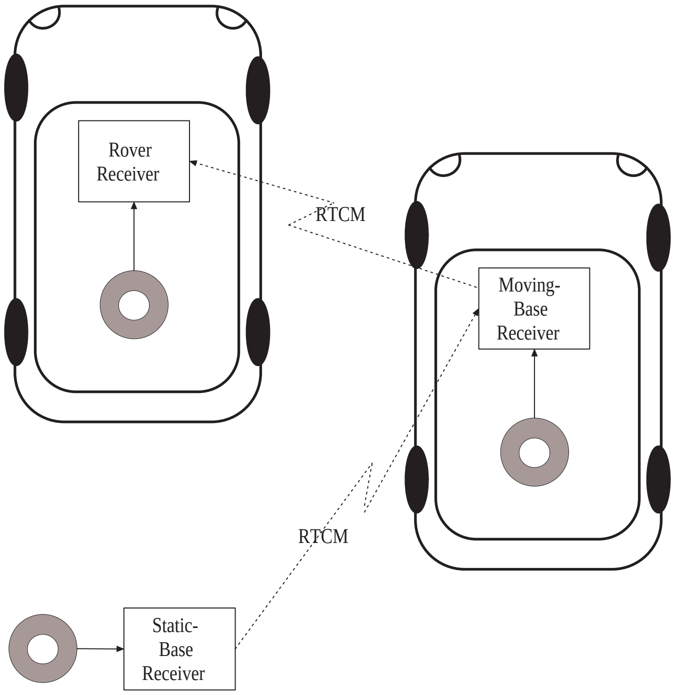

Figure 2-1: Example of a RTK moving-base configuration where the moving base receives RTCM corrections from a static base and transmits RTCM corrections to the rover.

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

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

1. The PVT engine must be set in one of the rover modes (standalone, DGPS, SBAS, RTK). The type of the PVT mode at the moving base station will determine the absolute position accuracy of the RTK rover receiver. On the other hand, the accuracy of the relative position of the rover with respect to the moving base is not influenced by the PVT mode at the moving base station. For instance, to let the moving base station compute a simple standalone PVT, use the following:
   `setPVTMode,Rover,StandAlone <CR>`
   RxControl: *Navigation > Positioning Mode > PVT Mode*

   If accurate absolute and relative positioning of the rover is required, the moving base can operate in RTK-rover positioning mode and receive RTCM or CMR corrections from a static base station, as illustrated in figure 2-1. Refer to section 2.7 to configure the moving-base receiver in RTK-rover mode.

2. From now on, follow the same procedure as for a static base station, starting at step 3 of section 2.5.1 and taking into account the following recommendations:
   - RTCM v2.x is not suited for moving-base operation, use only RTCM v3.x or CMR v2.0.
   - To decrease the effect of extrapolation errors, use a short RTCM or CMR message interval (see the commands **`setRTCMv3Interval`** and **`setCMRv2Interval`**). In most cases, it is safe to set the interval to its minimum value of 0.1 seconds. If the RTCM or CMR messages are sent through a COM connection, make sure that the baud rate is sufficient to support the high rate. A value of 115200baud is typical.

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

See also section 3.4.3.5 for more details on moving-base operation.

2.6 Configure the Receiver in DGPS-Rover Mode

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

Note that the rover requires at least RTCM message 1 to function in DGPS-rover mode.

In DGPS-rover mode, the base station must be static. Moving base stations are only supported in RTK-rover mode (see section 2.7).

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

1. The PVT processing needs to be configured to use DGPS corrections if they are available. If they are not available, your best choice would be to fall back on a standalone PVT. This can be configured by the following command:
   `setPVTMode,Rover,StandAlone+DGPS <CR>`
   RxControl: *Navigation > Positioning Mode > PVT Mode*

2. The connection from which to accept the RTCM messages has to be specified. This connection must be different from the one from which the commands are entered: the receiver will not accept commands from a port reserved for incoming RTCM data. For instance, assuming COM2 will be used for RTCM input:
   `setDataInOut,COM2,RTCMv2 <CR>`
   RxControl: *Communication > Input/Output Selection*

3. The baud rate has to be set to match the baud rate of the incoming RTCM stream. For instance if the incoming RTCM stream has a baud rate of 9600 baud, use:
   `setCOMSettings,COM2,baud9600 <CR>`
   RxControl: *Communication > COM Port Settings*

To go back to the standalone operation, type:
`setPVTMode,Rover,StandAlone <CR>`

For more details on DGPS refer to section 3.4.2.

2.7 Configure the Receiver in RTK-Rover Mode

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

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

1. The PVT processing needs to be configured to use RTK data if they are available. If they are not available, your best choice would be to fall back on a standalone PVT. This can be configured by the following command:
   `setPVTMode,Rover,StandAlone+RTK <CR>`
   RxControl: *Navigation > Positioning Mode > PVT Mode*

2. The connection from which to accept the RTCM messages or the CMR messages has to be specified. This connection must be different from the one from which the commands are entered: the receiver will not accept commands from a port reserved for incoming RTCM or CMR data. For instance, to tell the receiver to accept RTCM 2.x messages from its COM2 (only applicable to static base mode), use:
   `setDataInOut,COM2,RTCMv2 <CR>`

   To accept RTCM 3.x messages from COM2, use:
   `setDataInOut,COM2,RTCMv3 <CR>`

   If the RTK messages are sent in the CMR format, you should rather use:
   `setDataInOut,COM2,CMRv2 <CR>`
   RxControl: *Communication > Input/Output Selection*

3. The type of base station (static or moving) has to be specified. For a static base, you should use:
   `setDiffCorrUsage,,,,,off <CR>`
   For a moving base, use:
   `setDiffCorrUsage,,,,,on <CR>`
   RxControl: *Navigation > Positioning Mode > Differential Corrections*

4. The baud rate has to be set to match the baud rate of the incoming RTCM or CMR stream. For instance if the incoming RTCM stream has a baud rate of 9600 baud, use:
   `setCOMSettings,COM2,baud9600 <CR>`
   RxControl: *Communication > COM Port Settings*

   To go back to the standalone operation, type:
   `setPVTMode,Rover,StandAlone <CR>`

Please refer to section 3.4.3 for further details on the RTK positioning mode.

2.8 Determine the Attitude of a Vehicle

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

2.8.1 Moving-Base Attitude

As of firmware release 2.0, the heading and pitch of a vehicle can be derived from the orientation of the baseline between a base and a rover antenna when both antennas are attached to the vehicle. The base antenna is connected to a first receiver configured as RTK moving base station. The rover antenna is connected to a second receiver configured as RTK rover and accepting the RTCM stream from the first receiver. This is illustrated in figure 2-2.

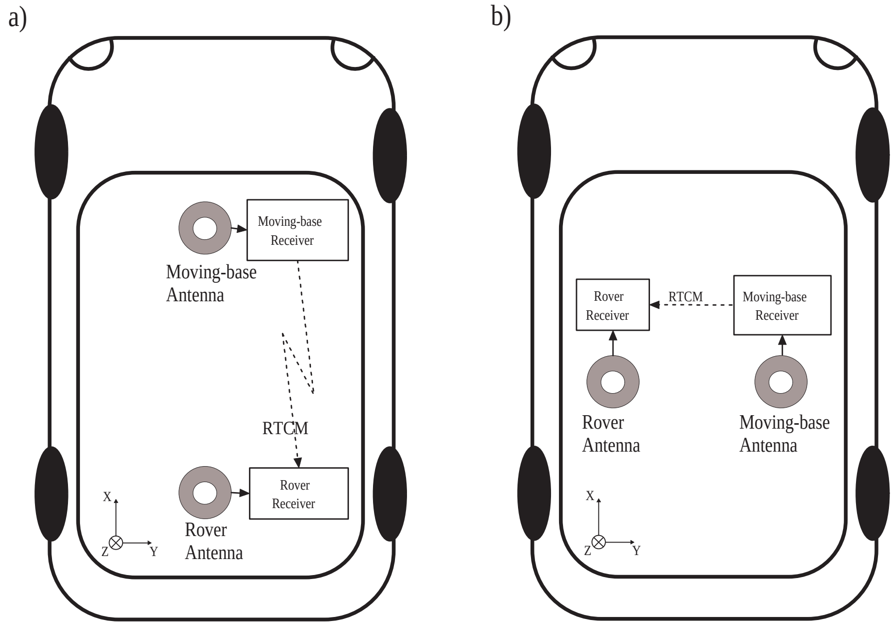

Figure 2-2: Moving-base attitude determination setup. a) default configuration. b) example of non-default configuration.

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

1. Attach two antennas to your vehicle. The default antenna configuration is as depicted in figure 2-2 a). It consists in placing the antennas aligned with the longitudinal axis of the vehicle. If such configuration is not possible, you will have to specify the relative position of your antennas, as explained below. For best accuracy, try to maximize the distance between the antennas.
2. Connect one of the antennas (preferably the one at the front of the vehicle) to the receiver that will serve as moving base. Connect the other to the receiver that will serve as rover. That latter receiver is the one where the heading and pitch will be computed.
3. Configure the moving-base receiver to send RTCM corrections to the rover. The procedure to do so is explained in section 2.5.2. Note that the RTCM stream may be transmitted either through a direct cable connection between the two receivers, or through a radio modem.

4. Configure the rover receiver to accept the RTCM corrections from the moving base, by following the steps in section 2.7.

5. By default, the attitude angles are computed assuming that the two antennas are aligned with the longitudinal axis of the vehicle, and that the moving-base antenna is in front of the rover antenna (see figure 2-2 a)). If you cannot place the antennas in such configuration, the reported attitude angles will be biased. That bias can be removed by telling the receiver where the moving-base antenna is located in the vehicle reference frame (see appendix A). This is done by specifying the coordinates of the baseline between the rover ARP and the moving-base ARP in the X, Y and Z directions. For example, in the configuration b) of figure 2-2, assuming that the distance between the antenna ARPs is 1 meter, you would issue (on the rover receiver):
   `setAntennaLocation, Base, manual, 0, 1, 0 <CR>`
   RxControl: *Navigation > Positioning Mode > GNSS Attitude*

6. Specify that the attitude has to be computed in moving-base mode by issuing the following command in the rover receiver:
   `setGNSSAttitude, MovingBase <CR>`
   RxControl: *Navigation > Positioning Mode > GNSS Attitude*

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

2.9 Track the GIOVE Satellites

If your receiver supports Galileo tracking, it will automatically track the signals from the GIOVE-A and -B satellites. GIOVE-A is tracked as Galileo PRN#32 (satellite code `E32` in commands like "**`setChannelAllocation`**") and GIOVE-B is tracked as Galileo PRN#31 (satellite code `E31`).

Because the receiver tracks GIOVE as Galileo PRNs 31 and 32, the genuine Galileo PRNs 31 and 32 can not be tracked. This is important to remember when using a Galileo constellation simulator. The support of GIOVE satellites will be discontinued at the end of the life of these satellites.

2.10 Configure the SBAS Operation

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

1. If you want to use the SBAS corrections to improve the PVT accuracy, you need to configure the PVT in SBAS mode. For instance, the following command instructs the receiver to compute a PVT using the SBAS corrections when available, and to fall back to the standalone mode otherwise:
   setPVTMode,Rover,StandAlone+SBAS <CR>
   RxControl: *Navigation > Positioning Mode > PVT Mode*

2. Make sure that the troposphere model is as prescribed by the RTCA DO 229 standard[2]. This is the default setting, but in case the receiver is not in its default configuration, you should use:
   setTroposphereModel,MOPS,MOPS <CR>
   RxControl: *Navigation > Receiver Operation > Position > Atmosphere*

3. It is recommended to leave the ionospheric model selection to `auto`. In particular, using the Klobuchar model in SBAS mode will lead to degraded performance and is not recommended.
   setIonosphereModel,auto <CR>
   RxControl: *Navigation > Receiver Operation > Position > Atmosphere*

4. By default, the receiver selects the SBAS satellite with the most SBAS corrections available. It is possible to force the receiver to select which SBAS satellite should provide the corrections to the PVT (and override the automatic selection by the receiver), and how to deal with subtleties of the SBAS navigation message. This is done by the **setSBASCorrections** command. For instance to only accept corrections from EGNOS PRN126, use:
   setSBASCorrections,S126 <CR>
   RxControl: *Navigation > Positioning Mode > SBAS Corrections*

5. Optionally, it is possible to include the range to SBAS satellites as an additional ranging source for the PVT. This is not done by default as the SBAS ephemeris accuracy is poor (100 m error). However to do so, use:
   setSatelliteUsage,+SBAS <CR>
   RxControl: *Navigation > Advanced User Settings > PVT > Satellite Usage*

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

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

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

For more details on SBAS positioning refer to section 3.4.1.

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

2.11 Generate a "Pulse Per Second" Signal

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

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

The command **`setPPSParameters`** can be used to synchronize the PPS with UTC, GLONASS or the internal time, or to alter the PPS interval and polarity. For instance, to synchronize the PPS with UTC and have one pulse every ten seconds, use:
`setPPSParameters, sec10, , ,UTC <CR>`
RxControl: *Navigation > Receiver Operation > Timing*

By default, the PPS pulse is calibrated so that it arrives at the right time (+/-10ns) at the PPS output port of the receiver when there is no antenna delays, no cable delays, and when the receiver is at a temperature of 20^{o}C. In an actual setup, the antenna and cable delays will cause the PPS to be offset from its correct position. The third argument of the **`setPPSParameters`** command can be used to specify the overall antenna and cable delay, in order to allow the receiver to compensate for them.

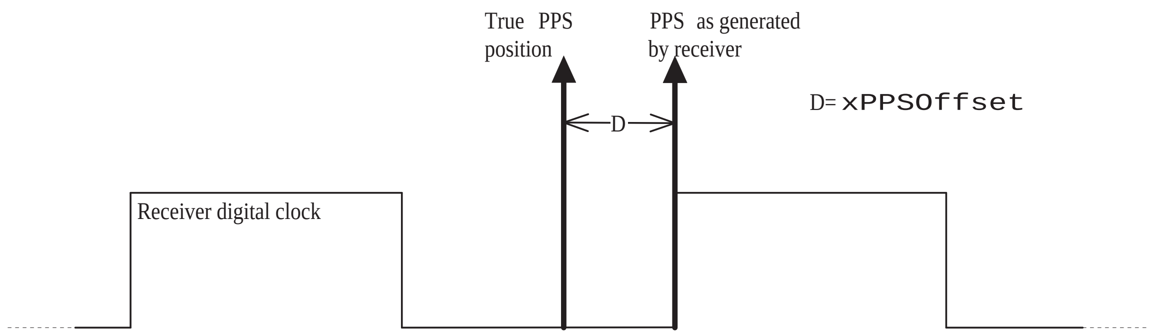

Figure 2-3: xPPS output granularity.

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

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

2.12 Time Tag External Events

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

By default, the receiver reacts on low-to-high transitions. You can use the **setEventParameters** command to react on falling edge instead:
setEventParameters, EventA, High2Low <CR>
RxControl: *Navigation > Receiver Operation > Timing*

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

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

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

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

2.13 Upgrade the Receiver

Upgrading the receiver is the process of installing a new GNSS firmware, a new FPGA configuration, a new permission file (see section 2.15) or a new antenna calibration file (see section 3.4.3.6).

Upgrading the GNSS firmware can clear the receiver configuration stored in non-volatile memory (see section 2.4). Please make sure to reconfigure your receiver (e.g. baud rate settings, elevation masks, LBAS1 access code, etc) after an upgrade.

There are several ways to upgrade the receiver:

1. By using the RxControl graphical interface (go to the *Tools* menu). Upgrading over USB is supported from firmware versions 2.3. Older versions only supported upgrading over serial ports.
2. By manually uploading upgrade files via one of the serial ports. This upgrade procedure is explained below.

Upgrade files are provided by Septentrio in two different formats: ".suf" and ".srec". The ".suf" file must be used for RxControl-based upgrades, while the ".srec" file must exclusively be used for the manual upgrade described below.

If you need to upgrade several components at once (e.g. the GNSS firmware and the FPGA configuration), you will need to repeat the upgrade procedure for each of the components. The following upgrade order is recommended: (1) GNSS firmware, (2) FPGA configuration, (3) permission file.

To manually upgrade the receiver, follow this procedure:

1. Connect to the receiver through one of its serial ports (only serial ports support the manual upgrade procedure).
2. Power cycle the receiver. When booting, the receiver outputs the following prompt:
   `$TE Septentrio SSRC5 SN <serialnr> is booting.\r\n`
   where `<serialnr>` is the serial number of your particular receiver.
3. After the above prompt is output, you have one second to break the automatic boot sequence. This is done by sending the following sequence of characters to the receiver:
   `GARx,saub <CR>`
4. If the boot is effectively interrupted, the receiver ouputs the `U-Boot>` prompt. At that prompt, enter the following command:
   `loads <CR>`
5. Transfer the ".srec" upgrade file in text mode to the receiver. Typically, on Windows, use Hyperterminal, select the *Transfer >Send Text File...* menu. The receiver outputs a series of dots, then a summary of the transfer.
6. When the file transfer is done, issue the following command to permanently write the data into the non-volatile memory of the receiver (select the command applicable to your particular receiver):
   On AsteRx1 receivers: **`autoscr 0x10000000 <CR>`**
   On all other receivers: **`autoscr 0x20000000 <CR>`**

7. The previous step can take several seconds. When it is completed, the receiver outputs the `U-Boot>` prompt. You can now power-cycle the receiver or reset it by entering:
   reset <CR>

8. The receiver restarts with the new firmware version. You can check the firmware version by entering the following command:
   lif, Identification <CR>

2.14 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 2.15.

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

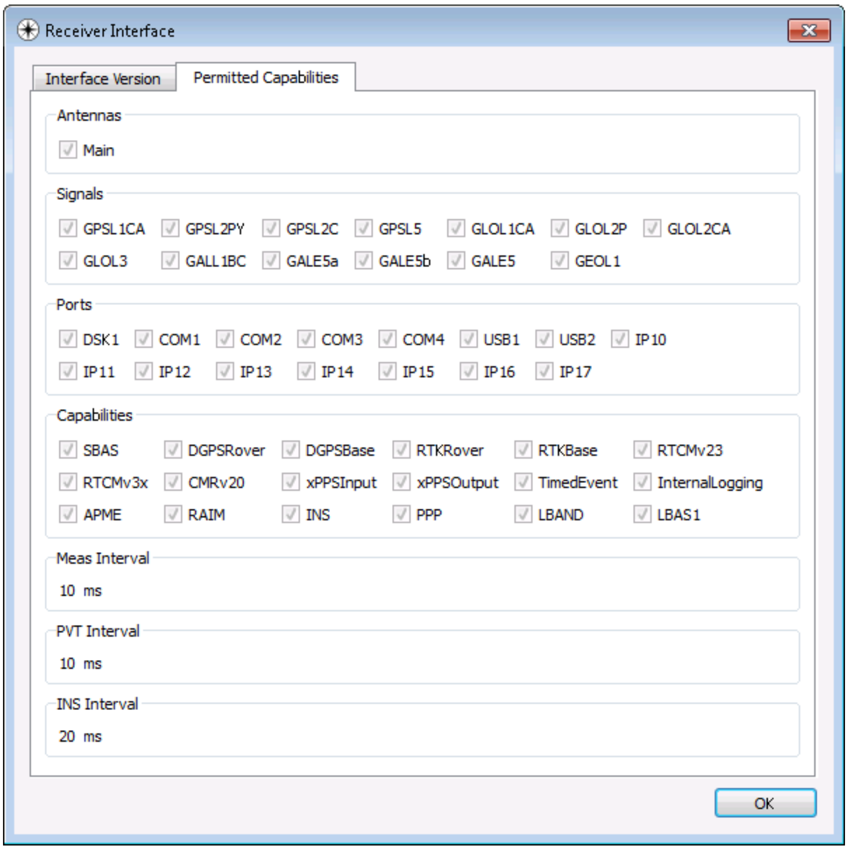

Figure 2-4: Example of receiver capabilities.

2.15 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 2.14.

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 2.13).

2.16 Manage the Processor Load

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

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

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

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

3 Operation Details

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

3.1 Channel Allocation and Signal Selection

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

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

3.2 Generation of Measurements

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

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

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

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

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

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

3.2.1 Pilot vs. Data Component

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

The table below indicates which signal component is used for all signals having a pilot and data component.

Signal	Signal component being used for measurement generation
Galileo L1	L1-C (for GIOVE satellites, L1-B is used instead)
Galileo E5a	E5a-Q
Galileo E5b	E5b-Q
Galileo E5AltBOC	E5AltBOC-Q
GPS L2C	L2C-L
GPS L5	L5-Q

3.3 Time Management

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

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

- Upon powering up the receiver, TOW and WNc are assumed unknown, and set to a "Do-Not-Use value" in the SBF blocks.
- The transmission time-of-week and week number are coded in the GPS or Galileo navigation messages:
  - As soon as the first time-of-week is decoded from the GPS or Galileo signal-in-space (SIS), the TOW counter is initialized to within 20 ms of GNSS system time and starts counting. This is also the time when the receiver starts generating measurements.
  - As soon as the week number is decoded from the GPS or Galileo SIS (which can be either simultaneously with the time-of-week, or several seconds later), the WNc counter is set and starts counting.
- After the first position and time fix has been computed (for which measurements from at least 4 satellites are required), TOW is set to within X milliseconds of GNSS time. This is done by introducing a jump of an integer number of milliseconds in the TOW counter. X is the maximal allowed offset between the receiver time and GNSS time, and is set by the **`setClockSyncThreshold`** command (by default, X=0.5ms). This initial clock synchronization leads to a simultaneous jump in all the pseudorange and carrier phase measurements.

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

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

3.3.1 Free-Running Clock

In free-running mode, the receiver time slowly drifts with respect to GNSS time. The receiver continuously monitors this time offset: this is the clock bias term computed in the PVT solution, as provided in the `RxClkBias` field of the `PVTCartesian` and `PVTGeodetic` SBF blocks. A clock jump of an integer number of milliseconds is imposed on the receiver clock each time the clock bias exceeds X milliseconds by an absolute value (X is set by **`setClockSyncThreshold`**). This typically results in a saw-tooth profile similar to that shown in Figure 3-1. In this example, X=0.5ms and each time the clock bias becomes greater than 0.5ms, a jump of 1ms is applied.

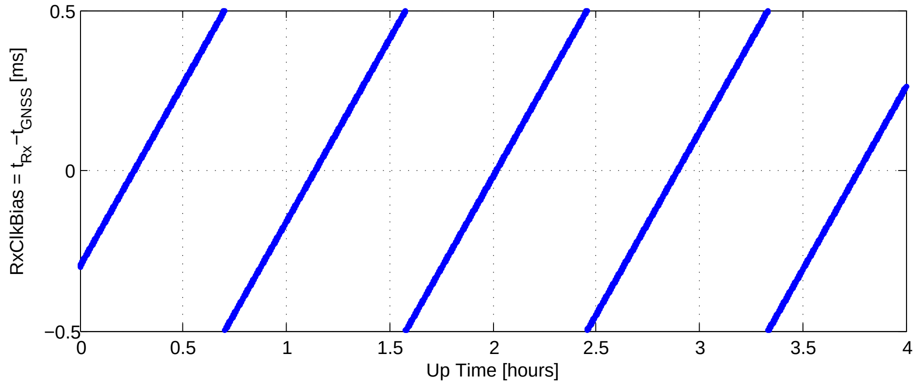

Figure 3-1: Example of the evolution of the receiver time offset with respect to the GNSS time in free-running mode.

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

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

3.3.2 Clock Steering

In steered mode, the receiver time is continuously steered to GNSS time to within a couple of nanoseconds. In the example of Figure 3-1, if the user would have enabled clock steering one hour after start up of the receiver, the clock bias would have been like in Figure 3-2 below.

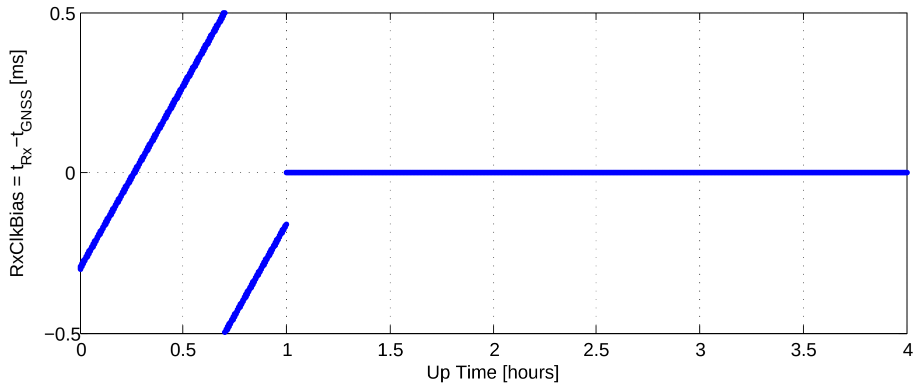

Figure 3-2: Effect of clock steering on the clock bias (clock steering enabled at an up time of 1 hour).

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

Note for the users of a GNSS constellation simulator

When using a constellation simulator, make sure to set the simulation time after January 01, 2006. The receiver time will be incorrect before that date.

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

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

The availability of the PVT depends on:

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

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

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

The accuracy of the PVT depends on:

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

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

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

3.4.1 SBAS Positioning

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

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

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

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

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

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

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

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

3.4.2 DGPS Positioning (Single and Multi-Base)

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

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

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

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

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

3.4.3 RTK Positioning

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

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

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

3.4.3.1 Pseudorange versus carrier phase: ambiguity

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

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

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

3.4.3.2 Carrier Phase Positioning

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

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

3.4.3.3 Integer Ambiguities (RTK-fixed)

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

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

3.4.3.4 Floating Ambiguities (RTK-float)

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

3.4.3.5 Moving Base

In RTK, the base station does not necessarily need to be static. In some applications, one is interested in the relative positioning of two moving vehicles. In that case, both base and rover receivers are mounted on moving platforms and the RTK engine computes the baseline between them. If both base and rover receivers are mounted on the same vehicle, the baseline can be used to determine the orientation of the vehicle (see section 2.8.1). If accurate absolute positioning is required in addition to relative positioning, the moving base receiver can operate in RTK mode and get RTK correction from a fixed base station (see section 2.5.2).

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

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

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

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

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

3.4.3.6 Antenna Effects

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

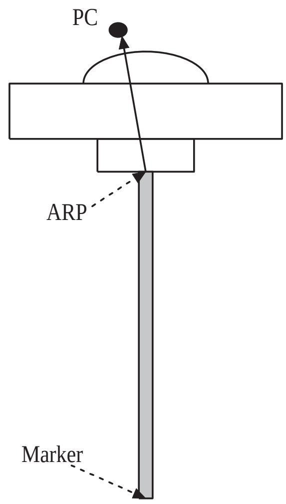

Figure 3-3: Antenna mount.

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

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

For high-precision positioning, the GNSS measurements need to be corrected in such a way that they all refer to a common and stable point in space. That point is referred to as the antenna reference point (ARP). For convenience, it is usually selected at the center of the bottom surface of the antenna. The National Geodetic Survey has calibrated the offset from the PC to the ARP as a function of the elevation and of the frequency band for a large number of geodetic-grade antennas. NGS publishes calibration tables that can be downloaded from the following URL:
`http://www.ngs.noaa.gov/ANTCAL/index.shtml`.
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 2.13 (the upgrade file is named `ant_info.suf`). To let the receiver compensate for the phase center variations and compute the ARP position, the user must specify the type of his/her antenna using the **setAntennaOffset** command. If the antenna is not specified, or the antenna type is not present in the antenna calibration file, the receiver cannot make the distinction between phase center and ARP, and the position accuracy is slightly degraded, especially in the height component.

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

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

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

3.4.3.7 Practical Considerations

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

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

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

The availability of fixed ambiguities increases significantly with the use of L2 carrier phase measurements. When in single-frequency operation, it is advised to force the receiver to remain in RTK-float mode, using the **setPVTMode** command.

3.4.4 Precise Point Positioning

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

PPP provides centimeter-level position accuracy, but suffers from a relatively long convergence time that can reach 15 to 20 minutes depending on the local multipath environment.

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

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

3.5 Receiver Autonomous Integrity Monitoring (RAIM)

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

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

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

The statistical tests assume an a-priori model of the measurement error probability distribution. As such, these tests can have the four classical outcomes in hypothesis testing, as shown in the table below (the letters A, B, C and D refer to the samples in Figure 3-4):

	no outlier	*outlier present*
outlier detected	False Alarm (type I error) A	Correct D
no outlier detected	Correct B	Missed Detection (type II error) C

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

The probability computations are based on the assumption that the residuals are distributed as a Normal distribution (central if there is no outlier, and non-central if there is one), as illustrated in Figure 3-4.

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

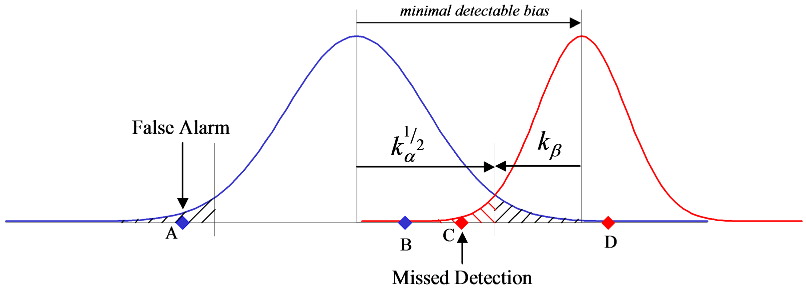

Figure 3-4: Statistical test outcomes.

Samples corresponding to the four test outcomes are represented in Figure 3-4: samples A and B are from the unbiased measurement distribution, while samples C and D are from a biased measurement distribution corresponding to an outlier. Since sample A is larger than the test threshold, it will be incorrectly flagged as an outlier (false alarm). Sample C is not detected as an outlier although it is part of the biased distribution (missed detection). The acceptable probability of false alarm and the probability of missed detection for the application must be determined and provided to the receiver. This is the purpose of the **`setRAIMLevels`** command.

3.5.1 Integrity Algorithm

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

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

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

where:

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

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

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

If the overall model test statistic is higher than the threshold, the test is rejected. In this case, the identification step will attempt to identify the measurement responsible for the rejection using the

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

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

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

where:

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

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

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

3.5.2 Internal and External Reliability Levels

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

The MDB describes the internal reliability of the corresponding w-test. It is a measure of the range error that can be detected with a given probability of missed detection. It is computed as follows for each satellite (neglecting the probability that the biased measurement falls on the left-hand side of the non-biased distribution shown in Figure 3-4):

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

where:

- σ_{y_i} is the standard deviation of the range measurement of the ith satellite;
- $\sigma_{\hat{y}_i}$ is the standard deviation of the estimator for the (measured) range of the ith satellite;

- λ_0 is the non-centrality parameter, which depends upon the probability of false alarm of the w-test and the probability of missed detection.

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

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

Detailed results of the RAIM algorithm are available in the `RAIMStatistics` and the `PVTResidual` SBF blocks and in the `GBS` NMEA message.

Appendix A Attitude Angles

A.1 Vehicle Reference Frame

The vehicle reference frame is attached to the vehicle. It has its X axis pointing along the longitudinal vehicle axis, the Y axis pointing towards the vehicle starboard (right) side and the Z axis pointing down, as illustrated in figure A-1.

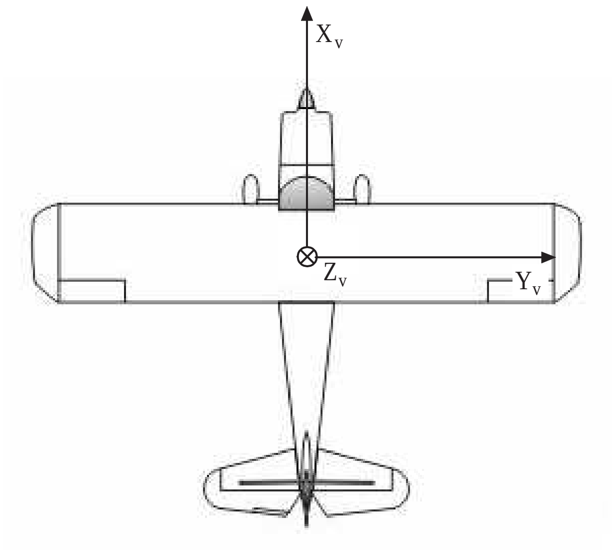

Figure A-1: Vehicle frame.

The attitude of the vehicle is defined as the angles between the vehicle frame and the local-level reference frame (defined by the East, North and Up directions). Septentrio receivers express the vehicle attitude in Euler angles using the heading-pitch-roll rotation sequence.

A.2 Euler Angles

Euler angles are defined as successive rotations of the vehicle frame (X, Y, Z axes) relative to the local-level East-North-Up reference frame. The rotation sequence is shown in figure A-2. 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;

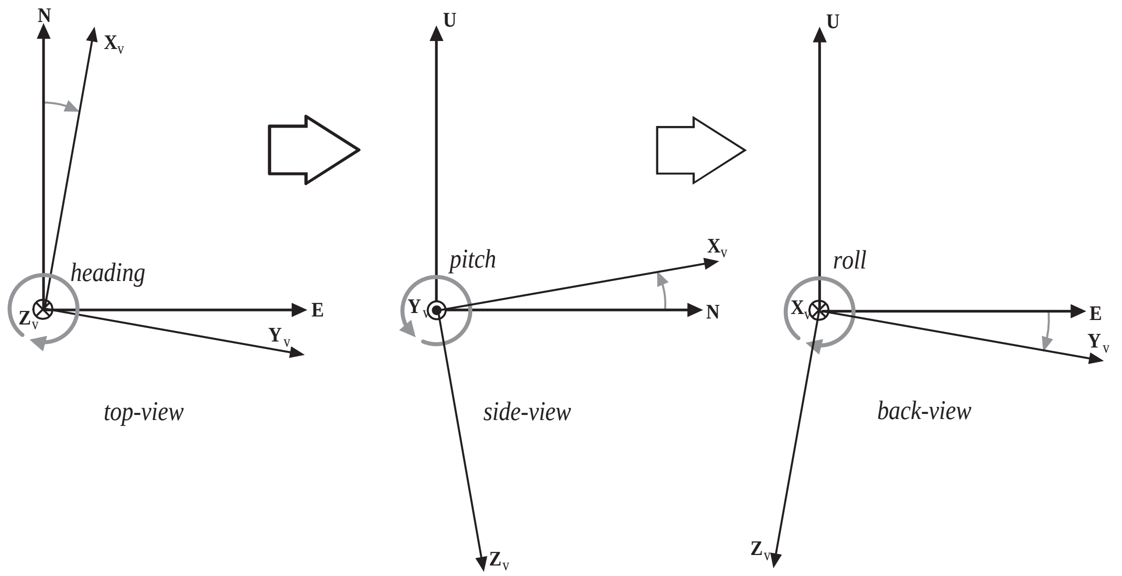

Figure A-2: Euler angle sequence.

Appendix B NMEA, RTCM and CMR Overview

The following tables provide a short overview of selected NMEA, RTCM and CMR messages. For a full description of these messages, please refer to the respective standard.

NMEA Sentence Type	Short Description
ALM	GPS Almanac Data
DTM	Datum Reference
GBS	GNSS Satellite Fault Detection
GGA	GPS Fix Data
GLL	Geographic Position - Latitude/Longitude
GNS	GNSS Fix Data
GRS	GNSS Range Residuals
GSA	GNSS DOP and Active Satellites
GST	GNSS Pseudorange Error Statistics
GSV	GNSS Satellites in View
HDT	Heading, True
RMC	Recommended Minimum Specific GNSS Data
ROT	Rate of Turn
VTG	Course Over Ground and Ground Speed
ZDA	Time and Date
LLQ	Leica Local Position and Quality
HRP	Heading, Roll, Pitch (Septentrio proprietary, see section B.1)
RBP	Rover-Base Position (Septentrio proprietary, see section B.1)
RBD	Rover-Base Direction (Septentrio proprietary, see section B.1)
RBV	Rover-Base Velocity (Septentrio proprietary, see section B.1)

CMR Message	Message Name
0	Observables
1	Reference Station Coordinates
2	Reference Station Description
3	GLONASS Observables

B.1 Proprietary NMEA Sentences

RTCM 2.x Message	Message Name
1	Differential GPS Corrections
3	GPS Reference Station Parameters
9	GPS Partial Correction Set
16	GPS Special Message
18	RTK Uncorrected Carrier Phases
19	RTK Uncorrected Pseudoranges
20	RTK Carrier Phase Corrections
21	RTK/Hi-Accuracy Pseudorange Corrections
22	Extended Reference Station Parameters
23	Antenne Type Definition Record
24	Antenna Reference Point (ARP)
31	Differential GLONASS Corrections
32	GLONASS Reference Station Parameters
59	Proprietary Message

RTCM 3.x Message	Message Name
1001	L1-Only GPS RTK Observables
1002	Extended L1-Only GPS RTK Observables
1003	L1&L2 GPS RTK Observables
1004	Extended L1&L2 GPS RTK Observables
1005	Stationary RTK Reference Station ARP
1006	Stationary RTK Reference Station ARP with Antenna Height
1007	Antenna Descriptor
1008	Antenna Descriptor and Serial Number
1009	L1-Only GLONASS RTK Observables
1010	Extended L1-Only GLONASS RTK Observables
1011	L1&L2 GLONASS RTK Observables
1012	Extended L1&L2 GLONASS RTK Observables
1013	System Parameters
1033	Receiver and Antenna Descriptors

PSSN,HRP	Septentrio Proprietary Sentence - Heading, Roll, Pitch

Field	Description
$PSSN,HRP,	Start of sentence
hhmmss.ss,	UTC of HRP (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Heading, degrees True
x.x,	Roll, degrees
x.x,	Pitch, degrees
x.x,	Heading standard deviation, degrees
x.x,	Roll standard deviation, degrees
x.x,	Pitch standard deviation, degrees
xx,	Number of satellites used for attitude computation
x,	Mode indicator: 0: No attitude available 5: Estimated attitude (dead-reckoning)
x.x,a	Magnetic variation, degrees (E=East, W=West, see also the **setMagneticVariance** command)
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBP	Septentrio Proprietary Sentence - Rover-Base Position

Field	Description
$PSSN,RBP,	Start of sentence
hhmmss.ss,	UTC of RBP (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	North (True) baseline component (positive when base is north of rover), meters
x.x,	East baseline component (positive when base is east of rover), meters
x.x,	Up baseline component (positive when base is higher than rover), meters
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c–c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBD	Septentrio Proprietary Sentence - Rover-Base Direction

Field	Description
$PSSN,RBD,	Start of sentence
hhmmss.ss,	UTC of RBD (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Azimuth of the base as seen from rover (0 to 360 increasing towards east), degrees True
x.x,	Elevation of the base as seen from rover (-90 to 90), degrees
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c–c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

PSSN,RBV	Septentrio Proprietary Sentence - Rover-Base Velocity

Field	Description
$PSSN,RBV,	Start of sentence
hhmmss.ss,	UTC of RBV (HoursMinutesSeconds.DecimalSeconds)
xxxxxx,	Date: ddmmyy
x.x,	Rate of change of baseline vector (rover to base), north component, m/s
x.x,	Rate of change of baseline vector (rover to base), east component, m/s
x.x,	Rate of change of baseline vector (rover to base), up component, m/s
xx,	Number of satellites used for baseline computation
x,	Quality indicator: 0: Invalid 2: DPGS 4: RTK 5: Float RTK
x,	Base motion indicator: 0: Static base 1: Moving base
x.x,	Correction Age, seconds
c–c,	Rover serial number
xxxx	Base station ID
*hh	Checksum delimiter and checksum field
<CR><LF>	End of sentence

Appendix C `sbf2rin` Utility

The CD-ROM accompanying your receiver contains the **sbf2rin** utility software.

sbf2rin converts a binary SBF file to the widely used RINEX ASCII format. RINEX v2.1 and 3.0 are supported. An SBF file is a file containing a succession of SBF blocks, possibly interspersed with other data (NMEA sentences for instance).

The following RINEX file types can be generated:

- Observation file (extension 'O');
- GPS navigation file (extension 'N');
- GLONASS navigation file (extension 'G');
- Galileo navigation file (extension 'L');
- SBAS navigation file (extension 'H');
- SBAS broadcast data (extension 'B').

In order to generate a RINEX file, the following procedure is recommended:

1. Use the **setAntennaOffset**, **setMarkerParameters** and **setObserverParameters** commands to specify the contents of the `ReceiverSetup` SBF block. The contents of this blocks is transferred to the RINEX header.

   The receiver has to be instructed to output the SBF blocks needed for the generation of the RINEX file (see section 2.3). The needed SBF blocks depend on the type of RINEX file:

RINEX file type	Mandatory and optional SBF blocks
Observation 'O'	`MeasEpoch` `PVTCartesian` or `PVTGeodetic` (optional: if not available, the "APPROX POSITION XYZ" line will be absent from the RINEX header) `ReceiverSetup` (optional: if not available, a default header will be generated, with most fields replaced by "unknown") `Comment` (optional: if available, user comments can be inserted in the RINEX file).
GPS Navigation 'N'	`GPSNav` `GPSIon` (optional: needed only if the header should contain the alpha and beta Klobuchar parameters) `GPSUtc` (optional: needed only if the header should contain UTC related data).
GLO Navigation 'G'	`GLONav` `GPSUtc` or `GALUtc` (this is mandatory : without at least one `GPSUtc` or `GALUtc` block in the file, **sbf2rin** is unable to generate a GLONASS navigation file).
Galileo Navigation 'L'	`GALNav` `GALIon` (optional) `GALUtc` (optional)
SBAS Navigation 'H'	`GEONav`
SBAS Broadcast 'B'	`GEORawL1`

2. Use RxControl or any suitable communication program to log the raw bytes coming from the receiver. Make sure that no character translation is applied by your logging program. Let's call the log file `LOG.SBF`. It is possible that `LOG.SBF` does not only contain SBF blocks, since the receiver may output other data in between two SBF blocks (replies to user commands, NMEA

sentences). This is not a problem: the SBF header allows identifying the SBF blocks in the raw stream from the receiver.

3. Use **sbf2rin** to generate a RINEX file from the log file `LOG.SBF`:
   sbf2rin -f LOG.SBF <CR>
   Note that the size of the SBF file must not exceed 2GBytes.

By default, **sbf2rin** generates a RINEX observation file. In order to generate the other file types, the **-n** option has to be used.

Invoking **sbf2rin** without argument prints the list of options and their usage.