



Hemisphere GPS Technical Reference v1.02

Table Of Contents

Introduction	13
GPS Technology and Platforms.....	14
GPS Engine.....	15
GPS Engine Overview	15
Satellite Tracking	15
Positioning Accuracy	15
Update Rates	16
DGPS Solutions	17
COAST Technology.....	17
SBAS - Overview	18
EGNOS.....	22
MSAS	22
GAGAN.....	23
Radiobeacon.....	23
OmniSTAR.....	25
Crescent Base Station Operation	28
e-Dif - Extended Differential Option for the Crescent Receiver	29
e-Dif Rover Mode Operation	30
e-Dif Startup.....	30
e-Dif Rover Calibration	30
e-Dif Rover Performance	30
L-Dif - Local Differential Option	31
L-Dif Startup.....	31
L-Dif Performance.....	31
RTK Overview	31

Post Processing	31
Hemisphere GPS Hardware Platforms	32
Crescent Vector OEM Development Kit.....	32
Evaluating Receiver Performance.....	33
Receiver Operation	35
Receiver Operation Overview	35
Communicating with the Receiver.....	35
Communicating with Receivers.....	35
NMEA 0183 Messages	35
Hemisphere GPS Proprietary Binary Interface	36
RTCM SC-104 Protocol	37
Firmware and Subscription Codes	38
Firmware.....	38
Subscription Codes.....	42
Configuring the Receiver.....	55
Configuring the Data Message Output.....	56
Saving the Receiver Configuration.....	57
Using Port D for RTCM Input	57
SBX-4 Database Mode.....	58
PocketMAX Utility	59
PocketMAX Overview.....	59
PocketMAX Key Uses	60
PocketMAX Startup	60
PocketMAX Features	61
PocketMAX GPS Tabs	61
Differential Source Tabs	61
TMNL Tabs.....	62

LOGS Tabs.....	63
HDG Tabs.....	63
Commands and Messages	65
NMEA 0183 Message Format	66
Command/Query/Message Types	67
General Operation and Configuration Commands.....	67
GPS Commands.....	69
SBAS Commands.....	69
e-Dif Commands.....	69
Crescent Vector Commands and Messages	70
GLONASS Commands and Messages.....	72
DGPS Base Station Commands.....	73
Local Differential and RTK Commands and Messages	73
Beacon Receiver Commands and Messages.....	74
NMEA 0183 SBX Queries.....	75
OmniSTAR Commands	75
RAIM Commands	77
Data Messages.....	77
Binary Messages	78
NMEA 2000 CAN Messages	79
Commands (All).....	80
GPCRQ,MSK Command	80
GPCRQ,MSS Command	81
GPMSK Command	82
JAGE Command.....	84
JAIR Command	85
JALT Command.....	87

JAPP Command	89
JASC Command Overview	91
JASC,CMR Command	92
JASC,D1 Command	93
JASC,DFX Command	94
JASC,GL Command	95
JASC,GN Command	96
JASC,GP Command	97
JASC,INTLT Command	99
JASC,PASHR Command	100
JASC,PSAT,RTKSTAT Command	102
JASC,PTSS1 Command	103
JASC,ROX Command	105
JASC,RTCM Command	106
JASC,RTCM3 Command	107
JASC,VIRTUAL Command	108
JATT Command Overview	109
JATT,CSEP Command	110
JATT,COGTAU Command	111
JATT,EXACT Command	112
JATT,FLIPBRD Command	113
JATT,GYROAID Command	114
JATT,HBIAS Command	115
JATT,HELP Command	116
JATT,HIGHMP Command	117
JATT,HRTAU Command	118
JATT,HTAU Command	119
JATT,LEVEL Command	120

JATT,MSEP Command	121
JATT,NEGILT Command	122
JATT,NMEAHE Command	123
JATT,PBIAS Command	124
JATT,PTAU Command	125
JATT,ROLL Command	126
JATT,SEARCH Command.....	127
JATT,SPDTAU Command	128
JATT,SUMMARY Command	129
JATT,TILTAID Command	131
JATT,TILTCAL Command	132
JBAUD Command	133
JBIN Command	135
JBOOT,OMNI Command	137
JCONN Command.....	138
JDIFF Command	139
JDIFFX,EXCLUDE Command	141
JDIFFX,GNSSOUT Command	142
JDIFFX,INCLUDE Command	144
JDIFFX,SOURCE Command.....	145
JDIFFX,TYPE Command.....	146
JFLASH Command Overview	147
JFLASH,DIR Command.....	148
JFLASH,FILE,CLOSE Command	149
JFLASH,FILE,NAME Command	150
JFLASH,FILE,OPEN Command	151
JFLASH,FREESPACE Command	152
JFLASH,NOTIFY,CONNECT Command	153

JFLASH,QUERYCONNECT Command	154
JFREQ Command	155
JGEO Command	157
JHP Command Overview	159
JHP,LIMIT Command	160
JHP,MODE,AUTOSEED Command	161
JHP,MODE,IGNORECONV Command	162
JHP,POS Command	163
JHP,POS,LAT,LON,HGT Command	164
JHP,POS,LAT,LON,HGT,,,OTHER Command	165
JHP,POS,OTHER Command	166
JHP,POS,PRESENT Command	167
JHP,RESET,ACCURACY Command	168
JHP,RESET,ENGINE Command	169
JHP,SEED Command	170
JHP,SEED,LAT,LON,HGT Command	171
JHP,STATIC Command	172
JHP,STATUS,AUTOSEED Command	173
JI Command	174
JK Command	175
JLBEAM Command	177
JLIMIT Command	179
JLXBEAM Command	180
JMASK Command	182
JMODE Overview	183
JMODE Command	184
JMODE,FOREST Command	185
JMODE,GPSONLY Command	186

JMODE,L1ONLY Command	187
JMODE,MIXED Command	188
JMODE,NULLNMEA Command	189
JMODE,SBASR Command	190
JMODE,TIMEKEEP Command.....	191
JMODE,TUNNEL Command	192
JMSG99 Command	193
JNMEA,GGAALLGNSS Command.....	194
JNMEA,PRECISION Command	195
JNP Command	196
JOFF Command	197
JOFF,ALL Command.....	198
JOMS Command	199
JPOS Command.....	200
JQUERY,GUIDE Command	201
JQUERY,RTKSTAT Command	202
JRAIM Command	204
JRAD Command Overview	205
JRAD,1 Command.....	206
JRAD,1,LAT,LON,HEIGHT Command	207
JRAD,1,P Command	208
JRAD,2 Command.....	209
JRAD,3 Command.....	210
JRAD,7 Command.....	211
JRAD,9,1,1 Command.....	212
JRELAY Command.....	213
JRESET Command	214
JRTK Command Overview	215

JRTK,1 Command	216
JRTK,1,LAT,LON,HEIGHT Command	217
JRTK,1,P Command	218
JRTK,5 Command	219
JRTK,5,Transmit Command	220
JRTK,6 Command	221
JRTK,12 Command	222
JRTK,17 Command	223
JRTK,18 Command	224
JRTK,28 Command	225
JSAVE Command	226
JSHOW Command	227
JSMOOTH Command	229
JT Command	230
JTAU Command Overview	231
JTAU,COG Command	232
JTAU,SPEED Command	233
JWAASPRN Command	234
PCSI,0 Command (Receiver Help Query command)	235
PCSI,1 Command (Status Line A, Channel 0 command)	236
PCSI,1,1 Command (Beacon Status command)	238
PCSI,2 Command (Status Line B, Channel 1 command)	239
PCSI,3,1 Command (Receiver Search Dump command)	241
PCSI,3,2 Command (Ten Closest Stations command)	242
PCSI,3,3 Command (Station Database command)	243
Messages (All)	244
Binary Messages Code	244
Bin1 Message	258

Bin2 Message	260
Bin62 Message	262
Bin65 Message	263
Bin66 Message	264
Bin69 Message	266
Bin76 Message	267
Bin80 Message	271
Bin89 Message	272
Bin93 Message	273
Bin94 Message	275
Bin95 Message	277
Bin96 Message	278
Bin97 Message	280
Bin98 Message	282
Bin99 Message	283
CRMSK Message	285
CRMSS Message	286
GLMLA Message	287
GNSSPositionData Message.....	289
GNSSPositionRapidUpdates Message.....	292
GPALM Message.....	293
GPDTM Message	294
GPGGA Message	295
GPGLL Message	297
GPGPS Message	298
GPGRS Message	300
GPGSA Message	301
GPGST Message.....	302

GPGSV Message	303
GPHDG/HEHDG Message	304
GPHDM/HEHDM Message.....	305
GPHDT/HEHDT Message	306
GPHEV Message.....	307
GPRMC Message.....	308
GPROT/HEROT Message.....	309
GPRRE Message	310
GPVTG Message.....	311
GPZDA Message.....	313
NMEACogSogData Message	314
PASHR Message.....	315
PSAT,GBS Message	316
PSAT,HPR Message	317
PSAT,INTLT Message.....	318
PSAT,RTKSTAT Message	319
RD1 Message.....	321
TSS1 Message	323
Resources.....	325
Reference Documents.....	325
Websites.....	326
Troubleshooting	327

Introduction

The purpose of the GPS Technical Reference is to serve as a resource for software engineers and system integrators engaged in the configuration of GPS receivers. It may also be of use to persons with knowledge of the installation and operation of GPS navigation systems.

This reference covers features, commands, logs, and operating modes for a variety of Hemisphere GPS products: not all aspects described apply to all products.

Information is provided as follows:

- [GPS Technology and Platforms](#) provides information on the GPS engine, GPS solutions, and GPS platforms
- [Receiver Operation](#) introduces general operational features of the receiver, receiver operation modes, and default operation parameters
- [PocketMAX Utility](#) provides a short introduction to PocketMAX PC and PocketMAX and what you can use them for. For more detailed information on PocketMAX refer to the PocketMAX User Guide available from www.hemisphergps.com.
- [Commands and Messages](#) are grouped by their type (General, GPS, e-Dif, Data, RAIM etc.) and for each type the commands or messages are initially listed in a table with a brief description. The commands and messages are then described in detail each in separate topics.
- Resources provides resources for additional information
- [Troubleshooting](#) provides troubleshooting advice

Copyright Notice

Hemisphere GPS Precision GPS Applications

Copyright © Hemisphere GPS (2011). All rights reserved.

No part of this manual may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of Hemisphere GPS.

GPS Technology and Platforms

- GPS Engine
- DGPS Solutions
- E-Dif
- L-Dif
- RTK
- Post Processing
- Hemisphere GPS Hardware Platforms
- Evaluating Receiver Performance

GPS Engine

GPS Engine Overview

The GPS engine is always operating regardless of the DGPS mode of operation. The following sections describe the general operation of the receiver.

- [Satellite Tracking](#)
- [Positioning Accuracy](#)
- [Update Rates](#)

Both the GPS and SBAS operation of the receiver module features automatic operational algorithms. When powered for the first time, the receiver system performs a "cold start," which involves acquiring the available GPS satellites in view and the SBAS differential service. To do this, the receiver needs a compatible GPS antenna connected that offers a relatively clear, unobstructed view of the sky. While you can often achieve this indoors with an antenna placed against a window, you may need to place the antenna outside, for example on a roof or a short distance away from the building.

If SBAS is not available in a particular area, an external source of [RTCM SC-104](#) differential correction may be used. If an external source of correction data is needed, the external source needs to support an eight data bit, no parity and one stop bit configuration (8-N-1). See also [SBAS Overview](#).

Satellite Tracking

The receiver automatically searches for GPS satellites, acquires the signal, and manages the associated navigation information required for positioning and tracking. This is a hands-free mode of operation. Satellite acquisition quality is described as a signal-to-noise ratio (SNR) and the higher the SNR, the better the signal reception quality. SNR information is provided by the receiver through the use of NMEA 0183 data messages available via its multiple serial ports.

Positioning Accuracy

The receiver is a sub-meter product with 95% horizontal accuracy under ideal conditions.

To determine the positioning performance of the receiver, Hemisphere GPS gathers a 24-hour data set of positions in order to log the diurnal environmental effects and full GPS constellation changes. Data sets shorter than 24 hours tend to provide more optimistic results.

The horizontal performance specification of 95% accuracy is, as stated above, based on ideal conditions. In reality, obstruction of satellites, multipath signals from reflective objects, and operating with poor corrections will detract from the receiver's ability to provide accurate and reliable positions. Differential performance can also be compromised if the receiver module is used in a region without sufficient ionospheric coverage. Further, if external corrections are used, the baseline separation between the remote base station antennas can affect performance.

Since the receiver will be used in the real world, blockage of the line of sight to SBAS satellites is often inevitable. The COAST function provides solace from obstruction of any differential correction source (SBAS, Beacon, RTCM, OmniSTAR, RTK, e-Dif) for 30 to 40 minutes depending on the amount of tolerable performance drift. In fact, our receivers will COAST when differential correction is lost no matter what the differential source is: SBAS, Beacon, RTCM, OmniSTAR, RTK, or e-Dif.

The estimated positioning precision is accessible through the use of NMEA 0183 command responses as described [Commands and Messages](#).

Because the receiver cannot determine accuracy with respect to a known location in real time (so is traditionally performed in post-mission analyses), the precision numbers are relative in nature and are only approximates.

Update Rates

The update rate of each NMEA 0183 and binary message of the receiver can be set independently with a maximum that is dependant upon the message type. For example, some messages have a 1 Hz maximum while other messages have a 20 Hz maximum. The higher update rates, such as 20 Hz, are an option and can be obtained at an additional cost.

Higher update rates are valuable for applications where:

- Higher speeds are present such as in aviation
- You have manual navigational tasks such as in agricultural guidance
- You have an automated or autonomous navigational task such as in robotics or machine control

DGPS Solutions

COAST Technology

Crescent and Eclipse OEM boards feature Hemisphere GPS' exclusive COAST technology that enables Hemisphere GPS Crescent and Eclipse receivers to utilize old DGPS correction data for 40 minutes or more without significantly affecting positioning quality.

Note: Crescent refers to Crescent, Crescent Vector, and Crescent Vector II OEM boards. Eclipse refers to Eclipse and Eclipse II OEM boards.

When using COAST, these receivers are less likely to be affected by differential signal outages due to signal blockages, weak signals, or interference.

Note: To obtain a full set of SBAS corrections, the receiver must receive the ionospheric map over a period of a few minutes. After this, the receiver can "coast" until the next set of corrections has been received.

COAST technology provides the following benefits:

- Accurate and minimal position drift during temporary loss of differential signal corrections
- Sub-meter accuracy up to 40 minutes after differential signal loss
- Outstanding performance in environments where maintaining a consistent differential link is difficult
- It is standard with Crescent and Eclipse GPS receiver technology

SBAS - Overview

The following topics describe the general operation and performance monitoring of the Space-Based Augmentation System (SBAS) demodulator within the receiver module:

- [Automatic tracking](#)
- [Performance](#)
- [WAAS](#)
- [WAAS DGPS](#)
- [WAAS Signal Information](#)
- [WAAS Reception](#)
- [WAAS Coverage](#)

SBAS Automatic Tracking

The SBAS demodulator featured within the receiver automatically scans and tracks multiple SBAS satellite signals, as specified by the [JWAASPRN](#) command (defaulted to WAAS PRN 135 and 138, suitable for use in North America).

If the default satellites become disabled, the receiver automatically tracks different satellites. This automatic tracking enables you to focus on other aspects of your application rather than ensuring the receiver is tracking SBAS correctly.

The SBAS demodulator features two-channel tracking that enhances the ability to maintain acquisition on an SBAS signal satellite in regions where more than one satellite is in view.

This redundant tracking approach results in more consistent signal acquisition in areas where signal blockage of either satellite is possible.

SBAS Performance

SBAS performance is described in terms of bit error rate (BER). The SBAS receiver requires a line of sight to the SBAS satellite to acquire a signal.

The BER number indicates the number of unsuccessfully decoded symbols in a moving window of 2048 symbols. Due to the use of forward error correction algorithms, one symbol is composed of two bits. The BER value for both SBAS receiver channels is available in the [RD1](#) message.

A lower BER indicates data is being successfully decoded with fewer errors, providing more consistent throughput. The BER has a default no-lock of 500 or more. As the receiver begins to successfully acquire a signal, a lower BER results. For best operation, this value should be less than 150 and ideally less than 20.

SBAS broadcasts an ionospheric map on a periodic basis and it can take up to five minutes to receive the map on startup. Until it downloads the SBAS map the receiver uses the broadcast ionosphere model, which can result in a lower performance compared to when the map has been downloaded. This is the case for any GPS product supporting SBAS services.

WARNING: When the map has been downloaded, you may observe a position jump due to the potential difference between the GPS ionospheric model and the ionosphere SBAS map. To minimize the impact of this issue on the use of the receiver, wait up to five minutes before using the receiver or issue the [JQUERYGUIDE](#) command to 'ask' the receiver if it feels the performance will be sufficient for operation.

WAAS

The US Federal Aviation Administration developed the Wide Area Augmentation System (WAAS) to provide accurate positioning to the aviation industry. In addition to providing a high quality and accurate service for this industry, the service is available free of charge to civilians and markets in North America.

Other government agencies have developed similar WAAS-compatible systems for their respective geographic regions.

- Europe - the European Space Agency, the European Commission and [EUROCONTROL](#) jointly developed the European Geostationary Navigation Overlay Service (EGNOS)
- Japan - the MTSAT Satellite-based Augmentation System (MSAS) was developed by the Japan Civil Aviation Bureau (JCAB)
- India - the Airport Authority of India and the Indian Space Research Organization ([ISRO](#)) are deploying the GPS Aided Geo Augmented Navigation system (GAGAN)

These compatible augmentation systems fall into a broader category often referred to as Space Based Augmentation System (SBAS). The receiver is capable of receiving correction data from all WAAS-compatible SBAS.

WAAS DGPS

WAAS differential, and other compatible SBAS, use a state-based approach in their software architecture. These services take in reference data from a network of base stations and endeavor to model the sources of error directly, rather than computing the sum impact of errors upon observed ranges. The advantage of this approach is that the error source can be more specifically accounted for during the correction process.

Specifically, WAAS calculates separate errors for the following:

- Ionospheric error
- GPS satellite timing errors
- GPS satellite orbit errors

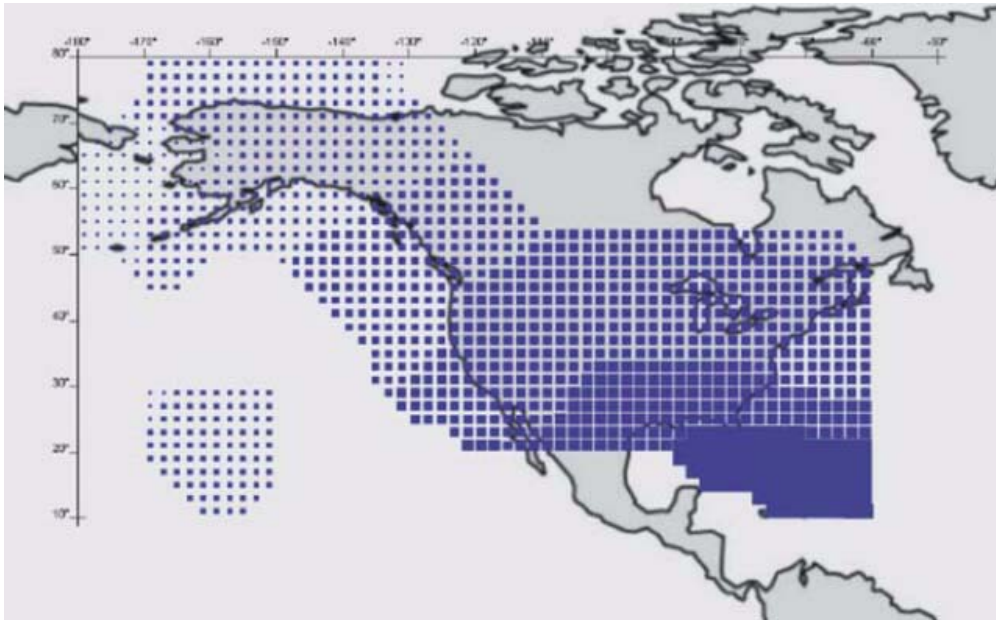
Provided that a GPS satellite is available to the WAAS reference station network for tracking purposes, orbit and timing error corrections will be available for that satellite. Ionospheric corrections for that satellite are only available if the signal passes through the ionospheric map provided by WAAS, which covers most of North America.

To improve the ionospheric map provided by WAAS, the receiver extrapolates information from the broadcast ionospheric coverage map, extending its effective coverage. This allows the receiver to be used successfully in regions that competitive products may not. This is especially important in Canada for regions north of approximately 54° N latitude and for outer regions of the Caribbean.

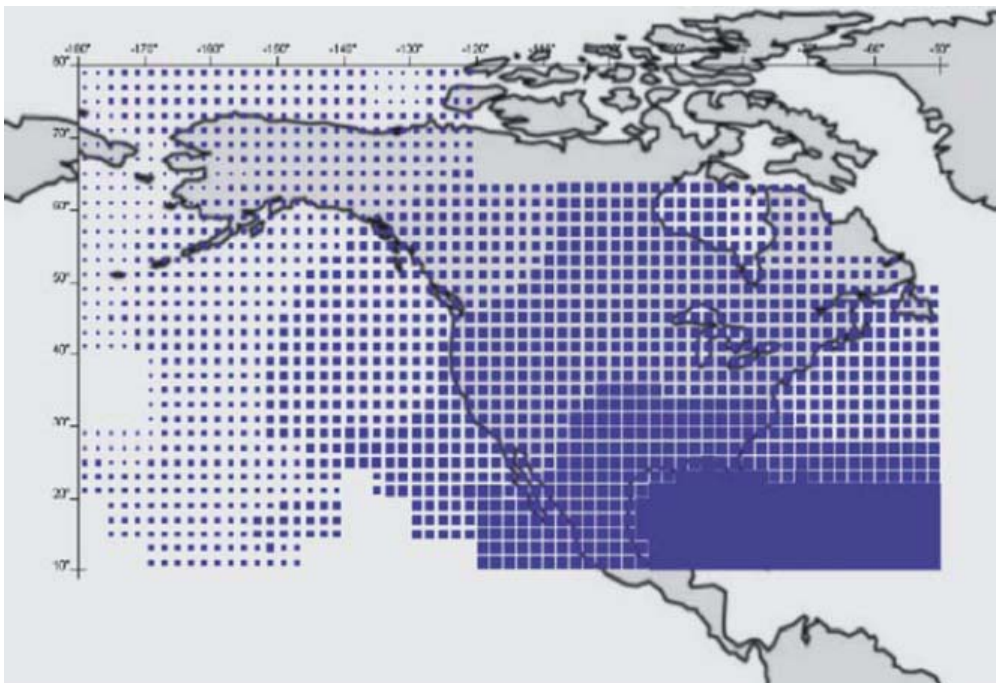
The process of estimating ionospheric corrections beyond the WAAS broadcast map is not as good as having an extended WAAS map and accuracy degradation may occur.

The map links below depict the broadcast WAAS ionospheric map coverage and the Hemisphere GPS extrapolated version, respectively. As the two maps show, the Hemisphere GPS extrapolated version's coverage is greater in all directions, enhancing usable coverage.

- Broadcast WAAS ionospheric correction map



- Extrapolated WAAS ionospheric correction map



WAAS Signal Information

WAAS and other SBAS systems transmit correction data on the same frequency as GPS, allowing the use of the same receiver equipment used for GPS. Another advantage of having WAAS transmit on the same frequency as GPS is that only one antenna element is required.

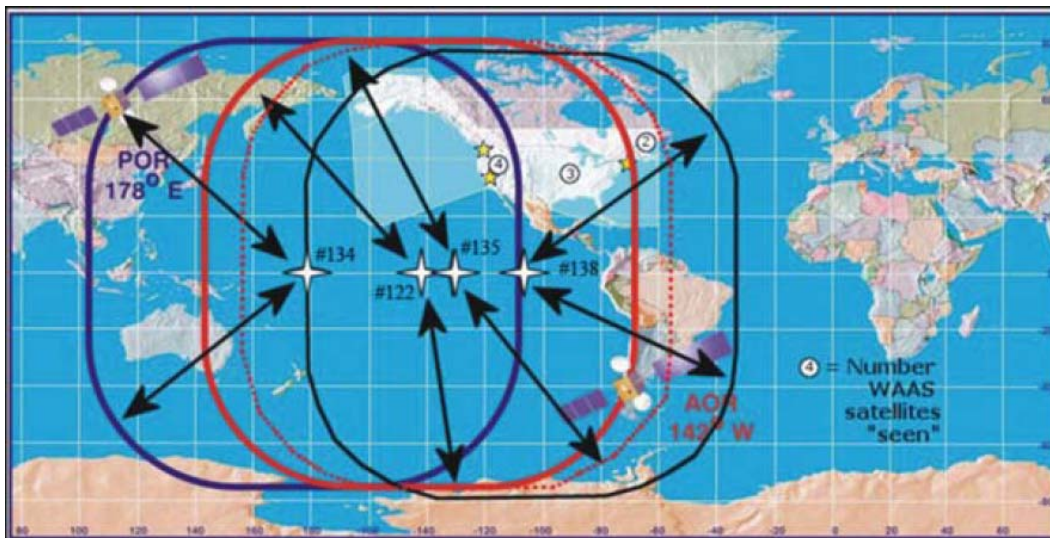
WAAS Reception

Since WAAS broadcasts on the same frequency as GPS, the signal requires a line of site in the same manner as GPS to maintain signal acquisition.

Because of their locations, SBAS satellites may appear lower on the horizon than GPS satellites—it depends on the geographic position on land. When using WAAS correction data, the receiver can provide the azimuth and elevation of all satellites to aid in determining their position with respect to the antenna.

WAAS Coverage

The figure below depicts the current WAAS coverage provided by the geostationary satellites.



The WAAS satellites are identified by their pseudorange number (PRN). In some areas, two or more satellites may be visible.

Note: Signal coverage may be present in some areas without either sufficient ionospheric map coverage or satellites with valid orbit and clock corrections. In such cases performance may be degraded compared to areas fully covered by the WAAS ionospheric coverage.

EGNOS

The European Geostationary Navigation Overlay Service (EGNOS) uses multiple geostationary satellites and a network of ground stations to transmit differential correction data for public use. EGNOS is currently located over the Atlantic Ocean and Africa.

Because of their location over the equator, these satellites may appear lower over the horizon as compared to GPS satellites - it depends on the geographic position on the land. In regions where the satellites appear lower on the horizon, they may be more susceptible to being masked by terrain, foliage, buildings or other objects, resulting in signal loss. Increased distance from the equator and the satellite's longitude cause the satellite to appear lower on the horizon. Hemisphere GPS's COAST technology helps alleviate this problem by maintaining system performance when EGNOS signal loss occurs for extended periods of time. More information on COAST technology is provided later in this chapter.

The figure below shows approximate EGNOS coverage provided by the satellites. Virtually all of Europe, part of Northern Africa, and part of the Middle East is covered with at least one signal. Most of Europe is covered by three signals.



Note: Increased distance from the equator and the satellite's longitude cause the satellite to appear lower on the horizon. Although a good amount of signal coverage is shown in northern latitudes for EGNOS, it may not be usable because of its low elevation angle and the potential for it to be obstructed. Testing of the system in the area of its use is recommended to ensure that the signal is sufficiently available.

MSAS

The MTSAT Satellite-based Augmentation System (MSAS) is currently run by the Japan Meteorological Agency (JMA). MSAS provides GPS augmentation information to aircraft through MTSAT (Multi-functional Transport Satellite) located approximately 36000 km above the equator (geostationary earth orbit).

MSAS generates GPS augmentation information by analyzing signals from GPS satellites received by monitor stations on the ground. This augmentation information consists of GPS-like ranging signal and correction information on GPS errors caused by the satellites themselves or by the ionosphere.

The MSAS signal provides accurate, stable, and reliable GPS position solutions to aircraft, resulting in a considerable improvement in the safety and reliability of GPS positioning. This enables aviation users who are under very strict safety regulations to use GPS positioning as a primary navigation system.

Visit <http://www.jma.go.jp/jma/jma-eng/satellite/> for more information on MSAS and MTSAT.

GAGAN

The GPS Aided Geo Augmented Navigation system (GAGAN) is currently under deployment by the Indian government and is anticipated to be operational by 2011. It operates similarly to the other SBAS regions described previously and will broadcast on one geostationary satellite (PRN 127) over the Western portion of the Indian Ocean. GAGAN should be visible in India at elevation angles in excess of 50° above the horizon. This will provide an excellent correction source in virtually all areas of the subcontinent.

Radiobeacon

Radiobeacon Overview

Many marine authorities, such as Coast Guards, have installed networks of radiobeacons that broadcast DGPS corrections to their users. With increasing use of these networks for terrestrial applications, there is increasing densification of these networks inland.

Radiobeacon Range

The broadcasting range of a 300 kHz beacon depends on a number of factors, including:

- Transmission power
- Free space loss
- Ionospheric state
- Surface conductivity
- Ambient noise
- Atmospheric losses

Signal strength decreases with distance from the transmitting station, mostly due to spreading loss. This loss is a result of the signal's power being distributed over an increasing surface area as the signal radiates away from the transmitting antenna.

The expected broadcast range also depends on the conductivity of the surface over which it travels. A signal will propagate further over a surface area with high conductivity than over a surface with low conductivity. Lower conductivity surfaces, such as dry, infertile soil, absorb the power of the transmission more than higher conductivity surfaces, such as sea water or arable land.

A radiobeacon transmission has three components:

1. **Direct line-of-sight wave**
The line-of-sight wave is insignificant beyond visual range of the transmitting tower and does not have a substantial impact upon signal reception.
2. **Ground wave**
The ground wave portion of the signal propagates along the surface of the earth, losing strength due to spreading loss, atmospheric refraction and diffraction, and attenuation by the surface over which it travels (dependent upon conductivity).
3. **Sky wave**
Depending on its reflectance, this skyward portion of the beacon signal may bounce off the ionosphere and back to Earth, causing reception of the ground wave to fade. Fading—which may cause reception to fade in and out—occurs when the ground and sky waves interfere with each other. This problem usually occurs in the evening when the ionosphere becomes more reflective and usually on the edge of coverage areas. Fading is not usually an issue with overlapping coverage areas of beacons and their large overall range.

Atmospheric attenuation plays a minor part in signal transmission range because it absorbs and scatters the signal. This type of loss is the least significant of those described.

Radiobeacon Reception

Various noise sources affect beacon reception and include:

- Engine noise
- Alternator noise
- Noise from power lines
- DC to AC inverting equipment
- Electric devices such as CRTs, electric motors, and solenoids

Noise generated by these types of equipment can mask the beacon signal, reducing or impairing reception.

Radiobeacon Antenna Location

When using the internal beacon receiver as the correction source, antenna location will influence the performance of the internal beacon receiver.

A good location will:

- Have a clear view of the sky (important for GPS, WAAS, and OmniSTAR signal reception)
- Be at least three feet away from all forms of transmitting antennas, communications, and electrical equipment, to reduce the amount of noise present at the antenna
- Be the best for the application, such as the center line of the vehicle or vessel (the position calculated by the beacon receiver is measured to the center of the antenna)
- Not be in areas that exceed specified environmental conditions

Radiobeacon Coverage

The figure below shows the approximate radiobeacon coverage throughout the world. Light shaded regions denote current coverage, with beacon stations shown as white circles. The world beacon networks continue to expand. For more current coverage, visit the Hemisphere GPS web site at www.hemispheregps.com.



OmniSTAR

OmniSTAR Overview

OmniSTAR is a worldwide terrestrial DGPS service that provides correction data to subscribers of the system with the use of a geostationary transponder.

The information broadcast by OmniSTAR DGPS is based on a network of reference stations—placed at geographically strategic locations—that communicate GPS correction data to control centers. At the control centers the GPS correction data is decoded, checked, and repackaged into a proprietary format for transmission to a geostationary L-band communications satellite. The satellite rebroadcasts the correction information back to earth over a large signal footprint where the Hemisphere GPS L-band differential satellite receiver demodulates the data.

The OmniSTAR signal content is not [RTCM SC-104](#), but a proprietary wide-area signal that's geographically independent. With this service, the positioning accuracy does not degrade as a function of distance to a base station because the data content is not composed of a single base station's information; it is composed of an entire network's information. When the Hemisphere GPS L-band DGPS receiver demodulates the proprietary signal it converts it into a local-area format for input to the GPS receiver (standard RTCM SC-104, message Type 1).

The L-band DGPS receiver interpolates corrections from the wide-area signal, specific to the location using Virtual Base Station (VBS) processing algorithms. The resulting RTCM corrections are those that would be calculated if a reference station were set up at the present location. This type of solution ensures a consistent level of accuracy across the entire coverage area. The GPS receiver provides position information to the L-band DGPS receiver for VBS calculations.

OmniSTAR offers three levels of service: VBS (described above), HP, and XP. HP and XP require a dual frequency receiver such as the Eclipse to function properly and are approximately three to seven times more accurate than the VBS service.

OmniSTAR Signal Information

The OmniSTAR L-band signal is a line-of-sight UHF signal that is similar to GPS. For the L-band differential receiver to acquire the signal, there must be a line of sight between the antenna and the geostationary communications satellite.

Various L-band communications satellites are used for transmitting the correction data to OmniSTAR users around the world. When the L-band receiver has acquired an OmniSTAR signal, the elevation and azimuth are available in the menu system to enable troubleshooting line-of sight problems.

[Contact OmniSTAR](#) for further information on this service.

OmniSTAR Reception

The OmniSTAR service broadcasts at a similar frequency to GPS and as a result is a line-of-sight system; there must be a line of sight between the antenna and the OmniSTAR satellite for reception of the service.

The OmniSTAR service uses geostationary satellites for communication. The elevation angle to these satellites is dependent upon latitude. For latitudes higher than approximately 55° North or South, the OmniSTAR signal may be blocked more easily by obstructions such as trees, buildings, and terrain.

OmniSTAR Coverage

The figure below shows approximate OmniSTAR service coverage. Regions without coverage, or with poor coverage, are shown with dark shading (Alaska, Northern Canada, Greenland, Iceland, and Northern Russia).



Note: Signal coverage may be present in some areas without reference stations within the region. Operating outside the reference station network may cause the applicability of the correction data to be less, resulting in a lower degree of positioning accuracy due to spatial decorrelation.

OmniSTAR Automatic Tracking

The Hemisphere GPS L-band DGPS receiver features an automatic mode that allows it to locate the best spot beam if more than one is available in a particular region. With this function you do not need to adjust the receiver's frequency. The receiver also features a manual tune mode for flexibility.

See the [JFREQ](#) command for more information on automatic and manual tuning.

OmniSTAR Receiver Performance

The OmniSTAR receiver provides both a lock indicator and a BER (Bit Error Rate) to describe the lock status and reception quality. Both these features depend on a line of sight between the antenna and the geostationary communications satellite broadcasting the OmniSTAR correction information.

OmniSTAR-capable Hemisphere GPS antennas is designed with sufficient gain at low elevation angles to perform well at higher latitudes where the signal power is lower and the satellite appears lower on the horizon. The BER number indicates the number of unsuccessfully decoded symbols in a moving window of 2048 symbols. Because of the use of forward error correction algorithms, one symbol is composed of two bits.

The BER has a default, no-lock value of 500. As the receiver begins to successfully acquire the signal a lower BER results. For best operation this value should be less than 150 and ideally less than 20.

OmniSTAR Subscription and Contact Information

OmniSTAR Service Activation

You can activate OmniSTAR DGPS service for a DGPS MAX receiver by contacting the service provider in the your region. Contact OmniSTAR with the unit number and OmniSTAR will activate the subscription over the air. Be prepared to have the receiver ready to receive the OmniSTAR signal for subscription validation.

OmniSTAR License Agreement

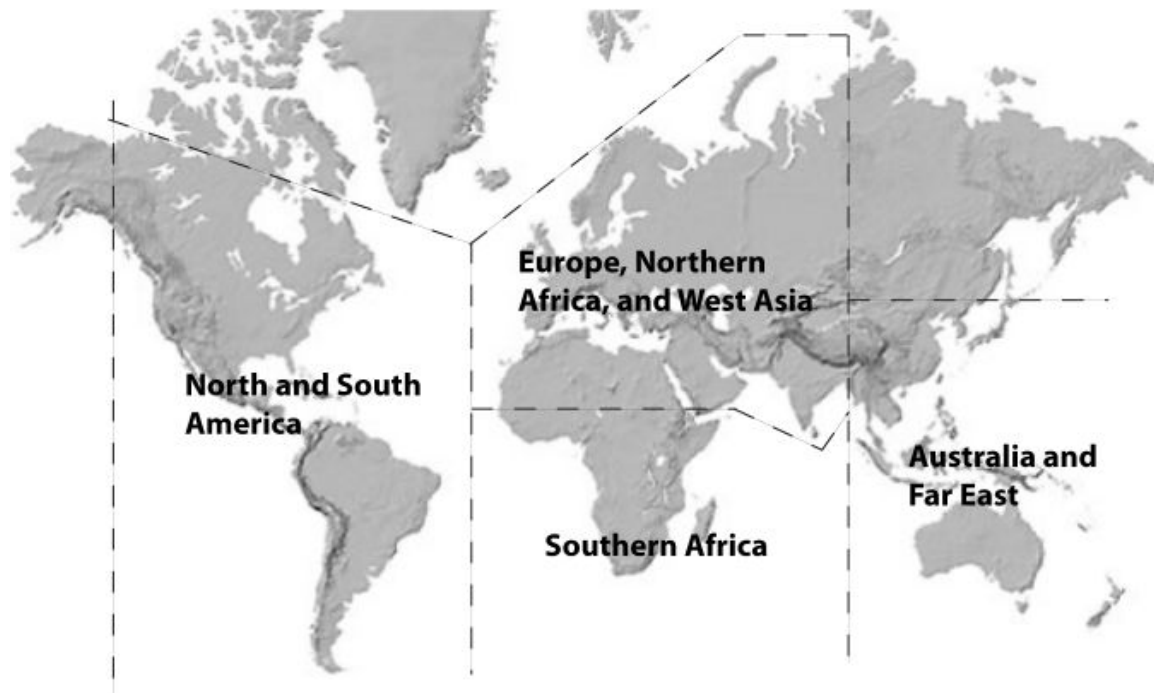
OmniSTAR requires that the enclosed license agreement be filled out the before subscription activation. Please read the agreement thoroughly before filling in the require information. Be ready to fax the completed agreement when contacting OmniSTAR.

Contacting OmniSTAR

Contact the office responsible for subscriptions in the your area based on the OmniSTAR coverage map below.

Visit www.omnistar.com for the most current contact information.

Location	Telephone Number	Website
North America South America	1-888-883-8476	www.omnistar.com
Europe Northern Africa Middle East West Asia	31-70-317-0900	www.omnistar.nl
Australia Far East	61-8-9322 5295	http://omnistar.com.au
Southern Africa	27 21 527 8950	www.omnistar.co.za



Crescent Base Station Operation

Crescent Base Station Overview

The Crescent receiver with e-Dif subscription can operate in a DGPS base station mode. NMEA 0183 commands need to be sent to the receiver to enter this mode. These commands may be automatically issued through customized software or through a simple terminal interface running on a PC, PDA, or data logger. [DGPS Base Station Commands](#) provides detailed information on the commands supported by the base station application.

Crescent Base Station Startup

When the receiver running the e-Dif application first starts up, it requires a few minutes to gather enough satellite tracking information to model the errors for the future. Once commands are sent to put the receiver into base station mode, corrections will be generated and can be sent via the serial port to rover receivers. In some more challenging GPS environments, the time required to model errors can take up to 10 minutes. The receiver must be stationary during this process and the antenna for the base station must be secured in a stable location.

Crescent Base Station Calibration

Base station calibration is the process of modeling the errors at the base station. Calibration can be performed in either a relative or an absolute sense, depending on positioning needs. Relative positioning provides positions that are accurate to one another but there may be some offset from the true geographical position.

Calibrating for relative positioning is easier than for absolute position since you are not restricted to using a point with known coordinates. Calibrating for absolute positioning mode requires placing the GPS antenna at a known reference location. Care should be taken to use a location that has good sky visibility and is relatively free from obstructions.

Crescent Base Station Performance

Base station performance depends primarily on the site location for the base station GPS antenna. An ideal location would have no obstructions above the height of the antenna, offering a full 180° by 360° view of the sky. In reality, obstructions such as trees, vehicles, people, and buildings nearby both block satellite signals and reflect interfering signals called multipath signals. Multipath degrades the accuracy of the satellite measurements and detracts from the receiver's ability to provide accurate and reliable corrections for the rovers.

For a rover to work optimally, a base station should be near by the rover's area of operation. As distance from the base to the rover increases, the modeling process cannot tune the solution to the exact environmental conditions at the rover's location and the rover's accuracy will not be as good. Best performance is attained when the distance from your base to your rover is less than 50 km (30 miles). Generally, there is little to no advantage to using a base station if it is more than 300 km (180 miles) from the rover.

e-Dif - Extended Differential Option for the Crescent Receiver

The Crescent receiver module is designed to work with Hemisphere GPS' patented Extended Differential (e-Dif) software. e-Dif is an optional mode where the receiver can perform with differential-like accuracy for extended periods of time without the use of a differential service. It models the effects of ionosphere, troposphere, and timing errors for extended periods by computing its own set of pseudo-corrections.

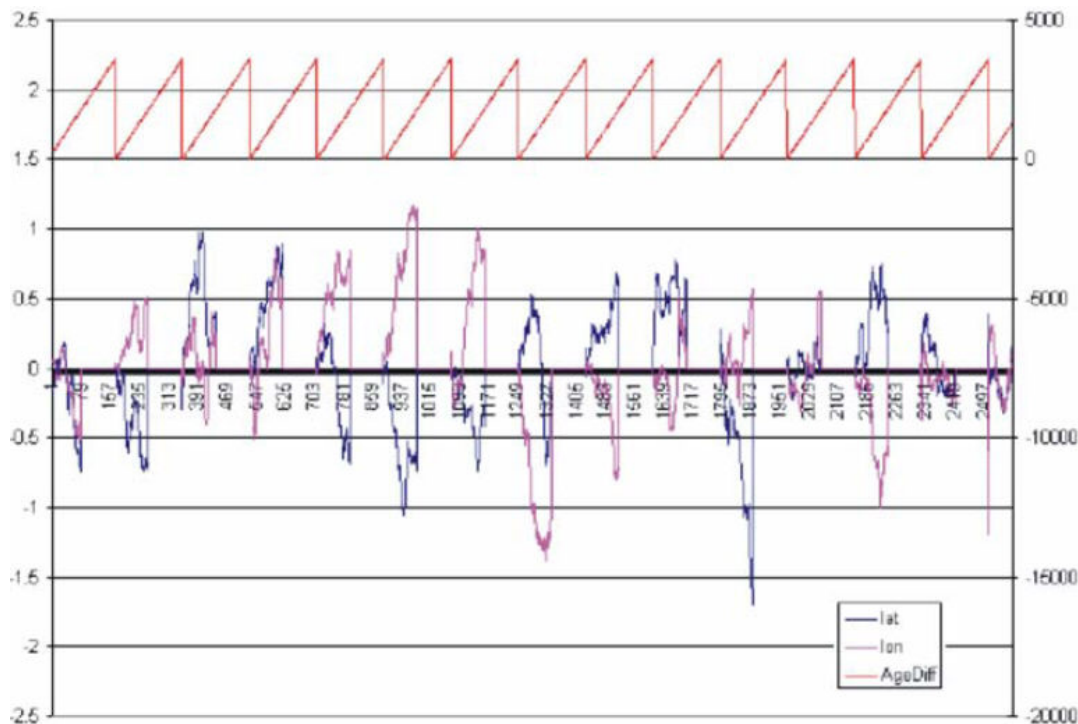
e-Dif may be used anywhere geographically and is especially useful where SBAS networks have not yet been installed, such as South America, Africa, Australia, and Asia. Two things are required to enable e-Dif. First your receiver will require the e-Dif application software to be installed on it. As well, a software key, called a subscription code, is needed for the receiver to use e-Dif. Both can be installed in the field using a PC computer. See [Using RightARM to Load Firmware](#) if you need to install the application firmware onto your receiver. To install a subscription code, contact Hemisphere GPS for a [JK command](#) which can be issued to your receiver.

Positioning with e-Dif is jump-free compared to a receiver working with just raw GPS provided the receiver consistently maintains a lock on at least four satellites at one time. The accuracy of positioning will have a slow drift that limits use of the e-Dif for approximately 30 to 40 minutes although it depends on how tolerant the application is to drift as e-Dif can be used for longer periods.

This mode of operation should be tested to determine if it is suitable for the application and for how long the user is comfortable with its use. As accuracy will slowly drift, the point at which to recalibrate e-Dif to maintain a certain level of accuracy must be determined.

The figure below displays the static positioning error of e-Dif while it is allowed to age for fourteen consecutive cycles of 30 minutes. The top line indicates the age of the differential corrections. The receiver computes a new set of corrections using e-Dif during the calibration at the beginning of each hour and modifies these corrections according to its models. After the initialization, the age correspondingly increases from zero until the next calibration.

The position excursion from the true position (the lines centered on the zero axis are northing [dark line] and easting [light line]) with increasing correction age is smooth from position to position; however, there is a slow drift to the position. The amount of drift depends on the rate of change of the environmental errors relative to the models used inside the e-Dif software engine.



Note: You decide how long e-Dif is to function before between calibrations and you should test this operation mode to determine an acceptable level of performance.

e-Dif Rover Mode Operation

Rover mode operation of the Crescent receiver unit with the optional e-Dif application requires NMEA 0183 commands. These commands may be automatically issued through customized software or through a simple terminal interface running on a PC, PDA or data logger. See [e-Dif Commands](#) for detailed information on the commands supported by the e-Dif feature.

e-Dif Startup

On startup, the receiver with the e-Dif application software running requires a few minutes to gather enough satellite tracking information to model the errors for the future. And in some environments this can take up to 10 minutes. The receiver does not have to be stationary for this process but it must be tracking the satellites throughout it. This process of gathering information and the subsequent initialization of e-Dif is referred to as "calibration."

e-Dif Rover Calibration

Rover calibration is the process of modeling the errors at the rover. Calibration can be performed in either a relative or an absolute sense, depending on positioning needs. Relative positioning provides positions that are accurate to one another but there may be some offset from the true geographical position. Additionally, unless the same point is used for all calibrations and its assumed position stored, it is possible for different cycles of e-Dif to have an offset.

Calibrating for relative positioning is easier than for absolute position, since you are not restricted to using a point with known coordinates. Calibrating for absolute positioning mode requires placing the GPS antenna at a known reference location. Use this point for subsequent calibrations.

e-Dif Rover Performance

The Crescent receiver's positioning performance is dependant upon the rate at which the environmental modeling of e-Dif and the environmental errors diverge. The more that e-Dif is able to model the errors correctly, the longer it will provide reliable and accurate positioning. As there is no way in real time to know the rate of divergence, a rule of thumb is to set the maximum age of differential to either 30 or 40 minutes, depending on how much error the application is able to tolerate (or simply recalibrate before 30 to 40 minutes goes by). Hemisphere GPS testing has shown that relative accuracy will often be better than 1.0 m 95% of the time after 30 minutes of e-Dif operation.

You should perform testing at your location to determine the level of performance that would be seen on average. When testing this feature, it is a good idea to look at a number of e-Dif cycles per day, and monitor performance against a known coordinate and possibly other receivers in autonomous and differential mode. You should do this over a number of days with different states of the ionosphere.

You can monitor the energy level of the ionosphere based upon the amount of solar flare activity at <http://www.spaceweather.com>.

L-Dif - Local Differential Option

Local differential (L-Dif) is a specialized message type that can be sent only between two Crescent-based receivers. One receiver is used as the base station and must remain stationary. It is extremely useful to know the coordinates of the base station position but averaging the position over several days will also suffice. The second receiver is used as a rover and the messages must be sent either through a cable or over a radio link.

L-Dif Startup

On startup, the receiver with the L-Dif running requires several [commands](#) to initialize the proprietary messages that are sent over the air.

L-Dif Performance

The receiver's positioning performance in L-Dif mode is dependant upon:

- Environment of the base and rover receivers
- Distance between them and
- Accuracy of the entered coordinates of the base station

Hemisphere GPS suggests you perform your own testing at your location to determine the level of performance you would expect on average. When testing this feature, conduct tests of 12-24 hours—in different environments—and monitor performance against a known coordinate. Do this over a number of days with different states of the ionosphere.

You can monitor the energy level of the ionosphere based upon the amount of solar flare activity at <http://www.spaceweather.com>.

RTK Overview

Real Time Kinematic (RTK) positioning is the highest form of navigational accuracy for GPS receivers. Hemisphere GPS offers RTK for both Crescent and Eclipse platforms. See [RTK commands](#) for more information.

Post Processing

Crescent and Eclipse receiver modules can output raw measurement data for post processing applications. The raw measurement and ephemeris data are contained in the [Bin 94](#) and [Bin 95](#) messages, and [Bin 96](#) (Crescent) or [Bin 76](#) (Eclipse) messages. All three messages must be logged in a binary file. Crescent receivers must log Bin 94, 95, and 96 messages, while Eclipse receivers must log Bin 94, 95, and 76 messages. Depending on the application, the binary data can be logged to a file and then translated to RINEX at a later time on a PC.

Hemisphere GPS provides a RINEX translator. It is available by contacting technical support at Hemisphere GPS; however, because there is limited ability to store station information in the binary file, developers may consider writing their own translator. Some code is available for developers but with very limited support. The code should be self-evident to developers familiar with RINEX and knowledgeable in C language.

Hemisphere GPS Hardware Platforms

Crescent Vector OEM Development Kit

Crescent Vector OEM Development Kit Overview

The purpose of the Crescent Vector OEM Development Kit is to provide accurate position and reliable heading information at high update rates. To accomplish this, the unit uses one high performance GPS engine and two multipath resistant antennas for GPS signal processing. One antenna is designated the primary GPS antenna and the other the secondary GPS antenna.

The unit computes the position by referencing the primary antenna center. It computes the heading by referencing the Vector baseline (formed by the distance between the primary and secondary antennas' centers).

Crescent Vector Calculations

The Crescent Vector's GPS engine uses both the L1 GPS C/A code and phase data to compute the location of the secondary GPS antenna in relation to the primary GPS antenna with a very high sub-centimeter level of precision. The technique of computing the location of the secondary GPS antenna with respect to the primary antenna, when the primary antenna is moving, is very similar to how "real time kinematic", or "RTK" solutions are computed. The primary antenna for the Crescent Vector operates in much the same way as the base antenna does for RTK.

RTK technology is very sophisticated and requires a significant number of possible solutions to be analyzed where various combinations of integer numbers of L1 wavelengths to each satellite intersect within a certain search volume. The integer number of wavelengths is often referred to as the "ambiguity," as they are ambiguous at the start of the RTK solution.

The Crescent Vector places a constraint on the RTK solution with the prior knowledge that the secondary GPS antenna has a fixed separation usually of 0.50 m (1.6 ft)—this can vary based on setup—from the primary GPS antenna. This considerably reduces the search volume, and therefore the startup times, because the location of the secondary antenna can theoretically fall only on the surface of a sphere with a radius of 0.50 m (1.6 ft) centered on the location of the primary antenna, versus a normal search volume that is greater than a cubic meter.

Supplemental Sensors

Supplemental Sensors - Reduced Time Search

In addition to incorporating the GPS engine, integrated inside the Crescent Vector are a gyro and a tilt sensor. When used, the tilt sensor aids the rate at which a heading solution is computed on startup and during reacquisition if the GPS heading is lost due to obstructions. Each supplemental sensor may be turned on or off individually; however, the full functionality of the Crescent Vector is realized only when all are used.

The tilt sensor further reduces the search volume from the volume associated with just a fixed antenna separation, because the Crescent Vector knows the approximate inclination of the secondary antenna with respect to the primary. The gyro only benefits reacquisition, because it initially requires a GPS heading to self-calibrate. The gyro further reduces the search volume.

Reducing the RTK search volume also has the benefit of improving the reliability and accuracy of selecting the correct heading solution by eliminating other possible erroneous solutions.

Note: Tilt and gyro aiding may be turned on depending on the product and may be disabled through user commands. Refer to your product's documentation for more information.

Supplemental Sensors - Heading System Backup

The Crescent Vector uses the gyro as a secondary source of heading for up to three minutes when there is a GPS outage due to obstruction. If the outage lasts more than three minutes, the gyro will be deemed to have drifted too far and will stop outputting. There is no user control over the timeout period of the gyro.

Evaluating Receiver Performance

Hemisphere GPS evaluates performance of the receiver with the objective of determining best-case performance in a real-world environment. Our testing has shown that the receiver achieves a performance better than 0.6 m 95% of the time in typical DGPS modes.

The qualifier of 95% is a statistical probability. Manufacturers often use a probability of RMS, one sigma, or one standard deviation. These three terms all mean the same thing and represent approximately 67% probability. Performance measures with these probabilities are not directly comparable to a 95% measure since they are lower probability (less than 70% probability).

Table 1 summarizes the common horizontal statistical probabilities.

Table 1: Horizontal Accuracy Probability Statistics	
Accuracy Measure	Probability (%)
rms (root mean square)	63 to 68
CEP (circular error probability)	50
R95 (95% radius)	95 to 98
2drms (twice the distance root)	95

It is possible to convert from one statistic to another using Table 2. Using the value where the 'From' row meets the 'To' column, multiply the accuracy by this conversion value.

Table 2: Accuracy Conversions				
	To			
From	CEP	rms	R95	2drms
CEP	1	1.2	2.1	2.4
rms	0.83	1	1.7	2.0
R95	0.48	.59	1	1.2
2drms	0.42	.5	.83	1

For example, Product A, after testing, has an accuracy of 90 cm 95% of the time (R95).

To compare this to Product B that has a sub-meter horizontal rms specification of 60 cm:

1. Select the value from where the 'R95' row and the 'rms' column intersect (to convert to rms). This conversion value is 0.59.
2. Multiply the 90 cm accuracy by this conversion factor and the result is 53 cm rms. Compared to Product B's 60 cm specification of sub-meter rms, Product A offers better performance.

To properly evaluate one receiver against another statistically, the receivers should be using identical correction input (from an external source) and share the same antenna using a power splitter (equipped with appropriate DC-blocking of the receivers and a bias-T to externally power the antenna). With this setup, the errors in the system are identical with the exception of receiver noise.

Although this is a comparison of the GPS performance qualities of a receiver, it excludes other performance merits of a GPS engine. The dynamic ability of a receiver should always be compared in a similar way with the test subjects sharing the same antenna. Unless a receiver is moving, its software filters are not stressed in a similar manner to the final product application. When testing dynamically, a much more accurate reference would need to be used, such as an RTK system, so that a "truth" position per epoch is available.

Further, there are other performance merits of a GPS engine such as its ability to maintain a lock on GPS and SBAS satellites. When evaluating this ability, the same GPS antenna should be shared between the receivers test subjects. For the sake of comparing the tracking availability of one receiver to another, no accurate "truth" system is required unless performance testing is also to be analyzed. Again, an RTK system would be required; however, it is questionable how its performance will fare with environments where there are numerous obstructions such as foliage. Other methods of providing a truth reference may need to be provided through observation times on surveyed monuments or traversing well-known routes.

Should you look to compare two RTK systems, determining truth can be very complicated. A rigorous dynamic comparison of two competing RTK systems should only be attempted by individuals and organizations familiar with RTK and potentially with inertial navigation equipment. Fortunately, most manufacturer's RTK performance is specified in similar accuracy values, and in general, RTK accuracy is quite similar across different manufacturers.

Note: Contact Hemisphere GPS technical support for further assistance in developing a test setup or procedure for evaluation of the receiver.

Receiver Operation

Receiver Operation Overview

When turned on, the receiver goes through an internal startup sequence. It is, however, ready to communicate immediately. Refer to the receiver-specific manual for the power specifications of the product.

When its antenna has an unobstructed view of the sky, the receiver provides a position in approximately 60 seconds and acquires SBAS lock in about 30 seconds more.

Note: The receiver can take up to 5 minutes to receive a full SBAS ionospheric map. Optimum accuracy is obtained when the receiver is processing corrected positions using complete ionosphere information.

Communicating with the Receiver

Communicating with Receivers

The receiver module features three primary serial ports (A, B, C) that may be configured independently of each other.

The ports can be configured to output a combination of data types:

- NMEA 0183
- Hemisphere GPS proprietary binary format
- [RTCM SC-104](#)

The usual data output is NMEA 0183 messages because these are the industry standard.

Note: If different data types are required to be output from the receiver simultaneously, such as NMEA 0183 and binary or NMEA 0183 and RTCM SC-104, ensure that the software used for logging and processing of the data has been designed to correctly parse the different data types from the single stream of data.

NMEA 0183 Messages

NMEA 0183 is a communications standard established by the National Marine Electronics Association (NMEA). NMEA 0183 provides data definitions for a variety of navigation instruments and related equipment such as gyrocompasses, Loran receivers, echo sounders, and GPS receivers.

NMEA 0183 functionality is virtually standard on all GPS equipment available. NMEA 0183 has an ASCII character format that enables the user to read the data via a receiving device with terminal software.

The following is an example of one second of NMEA 0183 data from the receiver:

```
$GPGGA,144049.0,5100.1325,N,11402.2729,W,1,07,1.0,1027.4,M,0,M,,010 *61
$GPVTG,308.88,T,308.88,M,0,0,0.04,N,0.08,K*42
$GPGSV,3,1,10,02,73,087,54,04,00,172,39,07,66,202,54,08,23,147,48,*79
$GPGSV,3,2,10,09,23,308,54,11,26,055,54,15,00,017,45,21,02,353,45*78
$GPGSV,3,3,10,26,29,257,51,27,10,147,45,45,,,,,,,,,*74
```

The NMEA 0183 standard allows manufacturers to define proprietary custom commands and to combine data into proprietary custom messages. Proprietary NMEA 0183 messages are likely to be supported only by specific manufacturers.

All messages and ports can be configured independently (see example below).

Port	Baud Rate	Messages
A	9600	GPGGA , one every 1 second GPGSV , one every 5 seconds
B	19200	GPGGA , one every 2 seconds Bin1 , one every 1 second Bin2 , one every 1 second

A selection of NMEA 0183 data messages can be configured at various update rates with each message having a maximum update rate. A different selection of NMEA 0183 messages with different rates can be configured on another port.

[Commands and Messages Overview](#) presents information about the NMEA 0183 interface of the receiver smart antenna.

See [Reference Documents](#) for contact information if you need to purchase a copy of the NMEA 0183 standard.

Hemisphere GPS Proprietary Binary Interface

Hemisphere GPS proprietary binary messages may be output from the receiver simultaneously with NMEA 0183 messages.

Binary messages are inherently more efficient than NMEA 0183 and would be used when maximum communication efficiency is required. Some receiver-specific pieces of information are only available through binary messages, such as raw data for post processing.

Note: If you need to log binary data, make sure the logging software has opened the file as a binary file; otherwise, data may be lost.

RTCM SC-104 Protocol

RTCM SC-104 is a standard that defines the data structure for differential correction information for a variety of differential correction applications. It was developed by the Radio Technical Commission for Maritime services (RTCM) and has become an industry standard for communication of correction information. RTCM is a binary data protocol and is not readable with a terminal program. Because it is a binary format and not ASCII text, it appears as "garbage" data on screen.

The following is an example of how the RTCM data appears on screen:

```
mRMP@PJfeUtNsmMFM{nVtIOTDbA^xGh~kDH`_FdW_yqLRryrDuh
cB\@}N`ozbSD@O^}nrGqkeTlpLLrYpDqAsrLRrQN{zW|uW@H`z]~aG
xWYt@|`_FxW_qqLRryrDCikA\@Cj]DE]]E@w_mIroMNjkKOsmMFM{
WDwW@HVEbA^xGhLJQH`_F`W_aNsmMFM[WVLA\@S}amz@illuP
qx~lZhTCpLLrYpdP@kOsmMFM[kVDHwVGbA^P{WWuNt_SW_yMs
mMnqdrhcC\@sE^ZfC@}vJmNGAHJVhTCqLRryrdviStW@H_GbA^
P{wxu[k
```

All Hemisphere GPS receivers support RTCM v2.x Type 1, Type 5, Type 6, and Type 9 messages for DGPS positioning.

Hemisphere GPS receivers do not support RTCM v2.x messages for RTK positioning. However RTCM v3.x messages (Type 1001 through 1008) are suitable for RTK positioning.

Note: RTCM v2.x is a local area data standard. This means that performance degrades as a function of distance from the base station when:

- Positioning with external connection input to the receiver from an external source or
- Outputting corrections from the receiver to another GPS receiver.

The additional degradation depends on the difference in observed orbit and ionospheric errors between the reference station and the remote unit. A general rule of thumb is an additional 1 m error per 100 miles.

This error is often seen as a bias in positioning, resulting in a position offset. The scatter of the receiver is likely to remain close to constant.

See [Reference Documents](#) for RTCM contact information to purchase a copy of the RTCM SC-104 specifications.

Firmware and Subscription Codes

Firmware

About Firmware

Hemisphere GPS products are built on one of three receiver platforms, each of which has specific firmware applications available.

- Crescent - WAAS, e-Dif, OmniSTAR VBS, L-Dif/RTK base, L-Dif/RTK rover
- Crescent Vector - WAAS, RTK rover
- Eclipse - WAAS/RTK base, RTK rover, OmniSTAR HP/XP

Some products may require purchasing a subscription code to unlock specific functionality. See [Subscription Codes](#) for more information.

As its name suggests, firmware is somewhere between hardware and software. Like software, it is a computer program which is executed by a microprocessor or a microcontroller. But it is also tightly linked to a piece of hardware, and has little meaning outside of it.


Within the context of GPS, the hardware is the GPS receiver and it is the receiver's processor that executes the firmware. The receiver's processor supports two simultaneous versions of firmware but only one version operates at a given time. The two versions—referred to as applications—may have different functionality. Use the [JAPP command](#) to change between two receiver applications.

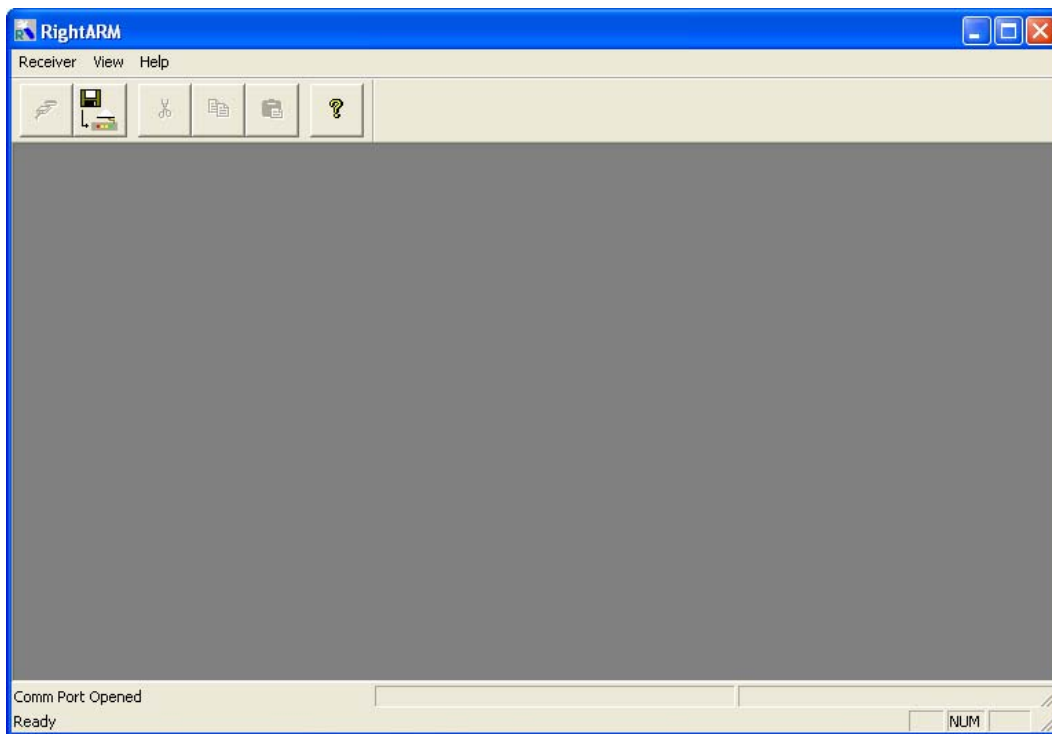
Using RightARM to Load Firmware

RightARM is Hemisphere GPS software that allows you to load the various GPS receiver firmware options and updates as they are provided by Hemisphere GPS.

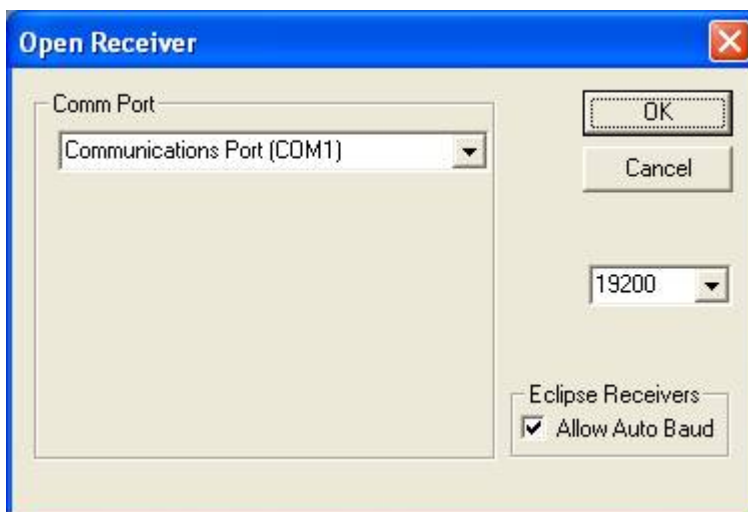
To load the firmware:

1. Download the latest version of RightARM from <http://www.hemispheregps.com>.
2. Install RightARM application on your computer.
3. Connect the receiver to your computer and power on the receiver.

4. Double-click the RightARM icon  to launch the program. The following screen appears.



5. Click the **Open Receiver** button  or select **Receiver > Connect**. The Open Receiver window appears, so you can identify a connected receiver.



6. Select the **Comm Port** on your computer to which the receiver is connected, select the 19200 baud rate for the receiver, and then click **OK**.

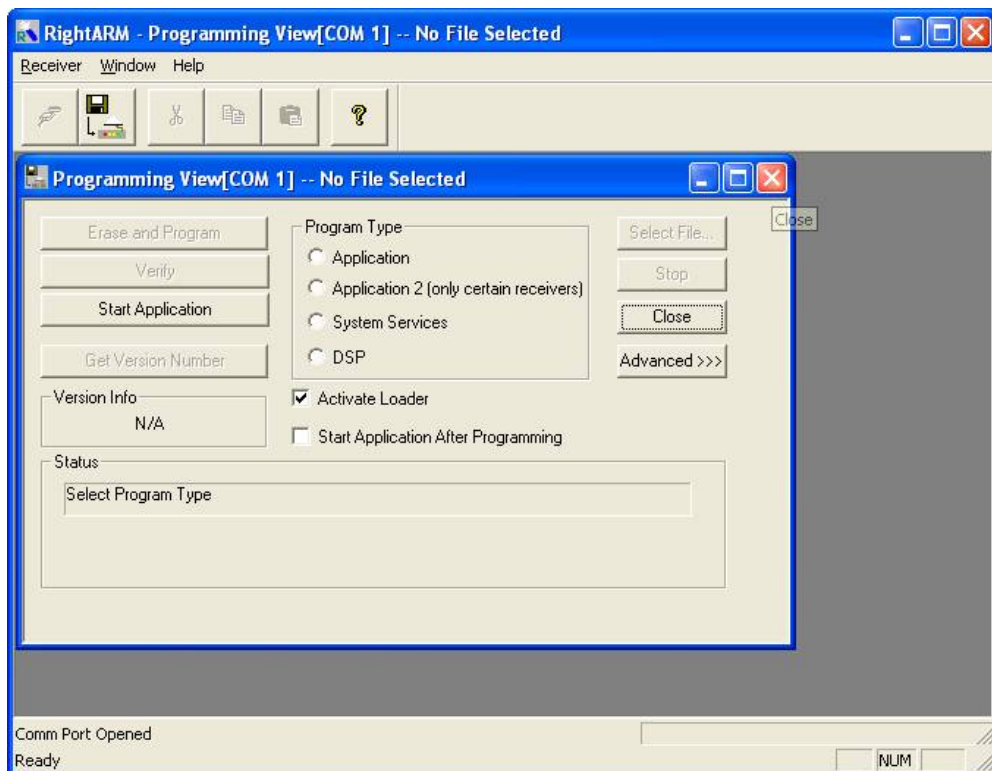
Note: You must set the baud rate to 19200.

When RightARM has successfully connected to the receiver the following message appears in the lower left corner of the screen.

Comm Port Opened
Ready



7. Click the **Programming View** button. The Programming View window appears, enabling you to select different firmware programming options.



8. Select the **Program Type** you want to install and then click **Select File**. The Open window appears.

Note: Most Hemisphere GPS receivers have two application locations available for firmware. In this case, select the Application option under Program Type and follow the remaining steps. Once the process is complete, you will repeat the process, selecting the Application 2 option when you reach this step again.

9. Select the required firmware file from the location where you saved it on your computer and click **Open**. "File Loaded" appears in the status window on the Programming View window.
10. Click the **Erase and Program** button to erase the firmware that is currently installed on the receiver in the selected application location and install the newly selected file in its place. "Erasing...Please Wait" appears in the Status field and a progress bar below this message indicates the programming progress. Once the new firmware has been successfully loaded on the receiver "Programming Done" appears in the Status field.

Note: Before pressing the Erase and Program button, the Activate Loader check box in the Programming View window will be selected. After pressing the Erase and Program button, the check box should be cleared and the Status field should show that the receiver is in loader mode and ready to receive the new firmware file. If the Activate Loader check box remains selected, turn the receiver off and then back on again, close and restart RightARM, and then start over at step 5.

WARNING: Do not interrupt the power supply to the receiver, and do not interrupt the communication link between the PC and the receiver until programming is complete. Failure to do so may cause the receiver to become inoperable and will require it to be returned to the factory for repair.

11. Once the appropriate firmware has been loaded, click the **Close** button to close the Programming View window.

Note: If a second application needs to be loaded, turn off the receiver, repeating all the steps starting at step 4, and on step 8 select the Application 2 option from the Program Type field.

12. Exit RightARM, turn off your receiver, and then disconnect the receiver from your computer.

Subscription Codes

Subscription Codes

Receiver activation has two steps:

1. Load application firmware
2. Enter the subscription code

This section covers:

- Finding the serial number and inputting a subscription code (e-Dif, L-Dif [base and rover], RTK, 20 Hz or 10Hz, etc.) into a Hemisphere GPS receiver
- Viewing the status and interpreting the \$JI subscription date codes
- The difference between the receiver's response to the [\\$JK](#) and [\\$JI](#) commands

Subscribing to an Application

Activating an application code on a Hemisphere GPS receiver requires the following:

- Serial communication cable to connect the Hemisphere GPS receiver to the serial COM port on the computer
- Download SLXMon from the www.hemisphergps.com and install on the computer or use a generic terminal program such as HyperTerminal
- Load the application to which to subscribe onto the Hemisphere GPS receiver (see [Using RightARM to Load Firmware](#))
- Purchase the application subscription code from Hemisphere GPS or an authorized Hemisphere GPS representative

To activate the application on a Hemisphere GPS receiver:

1. Connect the Hemisphere GPS receiver to the serial COM port on the computer.
2. Start SLXMon on the computer.
3. Select **File > Connect** and then select the appropriate Comm Port and Baud Rate to open communication with the receiver.
4. Select **Control > View Command Page**.
5. In the Receiver Command Page window type **\$JAPP** in the Message box and then click **Send**.
6. Confirm which applications are loaded onto the receiver and the order in which they appear in the Reply box.

Example Response (in Reply box):

```
$>JAPP,WAAS,DIFF
```

where WAAS (SBAS, EGNOS, MSAS) is the number one application (or application number 1) and DIFF (same as e-Dif) is the "other" application (or application number 2)

7. If DIFF is listed as application number 2 in the \$JAPP response then type the following command in the Message box:

```
$JAPP,O
```

where 'O' is the "other" application in the example. This swaps the two applications so that DIFF is be the current application.

8. Type the following command in the Message box:

```
$JI
```

The first number in the response is the serial number of the receiver.

Example Response (in Reply box):

```
$>JI,810133,1,3,09031998,01/06/1998,12/31/2018,3.5,31
```

The serial number is 810133. You will need to provide it to Hemisphere GPS with your request for an e-Dif subscription code.

9. Type the following command in the Message box after receiving the subscription code from Hemisphere GPS:

```
$JK,nnnn
```

where 'nnnn' is the subscription number. The receiver will respond with "subscription accepted."

Interpreting the \$JI and \$JK 'Date'/Subscription Codes

Subscriptions codes enable GPS differential correction sources on your receiver. When discussing them it is important to understand the following.

- The YYYY component of a MM/DD/YYYY formatted date—returned by both the [JI](#) and [JK](#) commands—is not always just the year component of that date. When a date's year starts with 30, only the 30 represents the year - and that year is 3000. A subscription expiration date of 01/01/3000 effectively means there is no expiration date.
- The last two digits of the 30YY 'date' represent the data output rate (in Hz) and the GPS differential correction sources that have been subscribed to and are therefore enabled on your receiver. Hemisphere GPS refers to these two digits as the "additive code" (see [Understanding Additive Codes](#)).
- The 30 and the 00 in the 'year' 3000, then, represents "Expires 3000 (so effectively does not expire), the data rate is 10 Hz, and SBAS is enabled." The 'year' 3015 indicates "Expires 3000, the data rate is 20 Hz and differential correction sources SBAS/e-Dif/RTK and L-Dif have been subscribed to and are enabled."

Below is an example of the \$JI command response, part of which is the subscription start and expiration dates (the date codes are shaded).

```
$>JI,12838,1,7,26022003,01/01/1900,01/01/3000,6.8Hz,38
```

Understanding Additive Codes

Tables 1 and 2 below provide subscription information for Crescent and Eclipse receivers, where the data rate and subscription are indicated by the 'date' returned by the [JK](#) and [JI](#) commands. For Eclipse II receivers, refer to [Eclipse II Subscription Codes](#). The part of the date that indicates the data rate and subscription code is called the "additive code." The last two digits in the subscription expiration date's 'year' comprise the additive codes, that is, the current data output rate from the receiver in Hz, plus the subscriptions—the enabled GPS differential correction sources.

Table 3 outlines the components of the Crescent, Eclipse, and Eclipse II additive codes. The subscription codes have different additive components for Crescent, Eclipse, and Eclipse II.

Table 1: Crescent Subscription Codes			
Date Code (Additive Code)	Hex Code	Maximum Data Rate	Subscription Description
3000 (0)	HEX 0	10 Hz	SBAS enabled
3001 (1)	HEX 1	20 Hz	SBAS enabled
3002 (0+2)	HEX 2	10 Hz	SBAS, e-Dif enabled
3003 (1+2)	HEX 3	20 Hz	SBAS, e-Dif enabled
3004 (0+4)	HEX 4	10 Hz	SBAS, RTK Rover enabled
3005 (1+4)	HEX 5	20 Hz	SBAS, RTK Rover enabled
3006 (0+2+4)	HEX 6	10 Hz	SBAS, RTK Rover, e-Dif enabled
3007 (1+2+4)	HEX 7	20 Hz	SBAS, RTK Rover, e-Dif enabled
3008 (0+8)	HEX 8	10 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Base enabled
3009 (1+8)	HEX 9	20 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Base enabled
3010 (0+2+8)	HEX A	10 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Base, e-Dif enabled
3011 (1+2+8)	HEX B	20 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Base, e-Dif enabled
3012 (0+4+8)	HEX C	10 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Rover, RTK Base enabled
3013 (1+4+8)	HEX D	20 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Rover, RTK Base enabled
3014 (0+2+4+8)	HEX E	10 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Rover, RTK Base, e-Dif enabled
3015 (1+2+4+8)	HEX F	20 Hz	SBAS, L-Dif Rover, L-Dif Base, RTK Rover, RTK Base, e-Dif enabled

Table 2: Eclipse Subscription Codes			
Date Code (Additive Code)	Hex Code	Maximum Data Rate	Subscription Description
3000 (0)	HEX 0	10 Hz	SBAS, OmniSTAR enabled
3001 (1)	HEX 1	20 Hz	SBAS, OmniSTAR enabled
3004 (0+4)	HEX 4	10 Hz	SBAS, OmniSTAR, RTK Rover, RTK Base, Raw L1/L2 data enabled
3005 (1+4)	HEX 5	20 Hz	SBAS, OmniSTAR, RTK Rover, RTK Base, Raw L1/L2 data enabled

Table 2: Eclipse Subscription Codes

Date Code (Additive Code)	Hex Code	Maximum Data Rate	Subscription Description
3008 (0+8)	HEX 8	10 Hz	SBAS, OmniSTAR, RTK Base, Raw L1/L2 data enabled
3009 (1+8)	HEX 9	20 Hz	SBAS, OmniSTAR, RTK Base, Raw L1/L2 data enabled
3016 (0+16)	HEX 10	10 Hz	SBAS, OmniSTAR, Raw L1/L2 data enabled
3017 (1+16)	HEX 11	20 Hz	SBAS, OmniSTAR, Raw L1/L2 data enabled

Eclipse II Subscription Codes (go [here](#))

Table 3: Crescent, Eclipse, and Eclipse II Additive Codes Components

Crescent		Eclipse		Eclipse II	
Code	Description	Code	Description	Code	Description
0	10 Hz	0	10 Hz	0	10 Hz
1	20 Hz	1	20 Hz	1	20 Hz
2	e-Dif	2	n/a	2	e-Dif
4	L-Dif Rover, L-Dif Base, RTK Rover	4	Raw L1/L2 Data, RTK Base, RTK Rover	4	RTK Rover (minimum L1 only)
8	RTK Base	8	Raw L1/L2 Data, RTK Base	8	RTK Base (minimum L1 only)
16	n/a	16	Raw L1/L2 Data	16	Raw Data (minimum L1 only)
32	n/a	32	n/a	32	L2 signals
64	n/a	64	n/a	64	GLONASS signals (minimum L1 only)

Crescent Additive Code Examples

- 10 Hz (SBAS), e-Dif, and RTK is $0+2+4 = 6$ (so 3006)
- 20 Hz (SBAS), e-Dif, and RTK is $1+2+4 = 7$ (so 3007)

Comparing the JI and JK Responses

In the following Crescent examples, the date code is shaded.

- Jl query date code example:
\$>JI,311077,1,7,04102005,01/01/1900,01/01/3000,6.8Hz,46
- JK date code example:
\$>JK,01/01/3000,0,(1, 2, 5 or no number)

In the JK examples, the second to last digit of the date code (,0, in the example) is the hex value (the second column of Table 2).

The last digit to the right (1, 2, 5 or no number) is the output rate in Hertz and indicates a downgrade from the default 10 Hertz. Thus, if 1, 2 or 5 does not appear, the output rate is the default 10 Hz.

The date codes are identical in either query and are directly related to each other. The last digit in the JK query is the hexadecimal equivalent of the last two digits in the date code. The following example further illustrate this. The date code is shaded.

Note: The JI response provides the decimal date code while the JK response provides both the decimal date code and the hex date code.

JI query date code example:

```
$>JI,311077,1,7,04102005,01/01/1900,01/01/3015,6.8Hx,46
```

JK date code example:

```
$>JK,01/01/3015,F
```

In this example, the date code is showing 15 in the last two digits. Therefore, the Hex number following the date code in the JK query is F as shown in the last row of Table 1.

Eclipse II Subscription Codes

Use the tables below locate your Eclipse II subscription code and its features.

1	2	4	8	16	32	64	
0x01	0x02	0x04	0x08	0x10	0x20	0x40	
20Hz	eDiff	RTK Rover, RTK Base, Raw Out	RTK Base, Raw Out	Raw Out	L2	GLONASS	Date Code (Additive Code)
							3000
Y							3001
	Y						3002
Y	Y						3003
		Y					3004
Y		Y					3005
	Y	Y					3006
Y	Y	Y					3007
			Y				3008
Y			Y				3009
	Y		Y				3010
Y	Y		Y				3011
		Y	Y				3012
Y		Y	Y				3013
	Y	Y	Y				3014
Y	Y	Y	Y				3015
				Y			3016
Y				Y			3017
	Y			Y			3018
Y	Y			Y			3019
		Y		Y			3020
Y		Y		Y			3021
	Y	Y		Y			3022
Y	Y	Y		Y			3023
			Y	Y			3024
Y			Y	Y			3025
	Y		Y	Y			3026
Y	Y		Y	Y			3027
		Y	Y	Y			3028
Y		Y	Y	Y			3029
	Y	Y	Y	Y			3030
Y	Y	Y	Y	Y			3031
					Y		3032
Y					Y		3033

1	2	4	8	16	32	64	
0x01	0x02	0x04	0x08	0x10	0x20	0x40	
20Hz	eDiff	RTK Rover, RTK Base, Raw Out	RTK Base, Raw Out	Raw Out	L2	GLONASS	Date Code (Additive Code)
	Y				Y		3034
Y	Y				Y		3035
		Y			Y		3036
Y		Y			Y		3037
	Y	Y			Y		3038
Y	Y	Y			Y		3039
			Y		Y		3040
Y			Y		Y		3041
	Y		Y		Y		3042
Y	Y		Y		Y		3043
		Y	Y		Y		3044
Y		Y	Y		Y		3045
	Y	Y	Y		Y		3046
Y	Y	Y	Y		Y		3047
				Y	Y		3048
Y				Y	Y		3049
	Y			Y	Y		3050
Y	Y			Y	Y		3051
		Y		Y	Y		3052
Y		Y		Y	Y		3053
	Y	Y		Y	Y		3054
Y	Y	Y		Y	Y		3055
			Y	Y	Y		3056
Y			Y	Y	Y		3057
	Y		Y	Y	Y		3058
Y	Y		Y	Y	Y		3059
		Y	Y	Y	Y		3060
Y		Y	Y	Y	Y		3061
	Y	Y	Y	Y	Y		3062
Y	Y	Y	Y	Y	Y		3063
						Y	3064
Y						Y	3065
	Y					Y	3066
Y	Y					Y	3067
		Y				Y	3068
Y		Y				Y	3069
	Y	Y				Y	3070

1	2	4	8	16	32	64	
0x01	0x02	0x04	0x08	0x10	0x20	0x40	
20Hz	eDiff	RTK Rover, RTK Base, Raw Out	RTK Base, Raw Out	Raw Out	L2	GLONASS	Date Code (Additive Code)
Y	Y	Y				Y	3071
			Y			Y	3072
Y			Y			Y	3073
	Y		Y			Y	3074
Y	Y		Y			Y	3075
		Y	Y			Y	3076
Y		Y	Y			Y	3077
	Y	Y	Y			Y	3078
Y	Y	Y	Y			Y	3079
				Y		Y	3080
Y				Y		Y	3081
	Y			Y		Y	3082
Y	Y			Y		Y	3083
		Y		Y		Y	3084
Y		Y		Y		Y	3085
	Y	Y		Y		Y	3086
Y	Y	Y		Y		Y	3087
			Y	Y		Y	3088
Y			Y	Y		Y	3089
	Y		Y	Y		Y	3090
Y	Y		Y	Y		Y	3091
		Y	Y	Y		Y	3092
Y		Y	Y	Y		Y	3093
	Y	Y	Y	Y		Y	3094
Y	Y	Y	Y	Y		Y	3095
					Y	Y	3096
Y					Y	Y	3097
	Y				Y	Y	3098
Y	Y				Y	Y	3099
		Y			Y	Y	3100
Y		Y			Y	Y	3101
	Y	Y			Y	Y	3102
Y	Y	Y			Y	Y	3103
			Y		Y	Y	3104
Y			Y		Y	Y	3105
	Y		Y		Y	Y	3106
Y	Y		Y		Y	Y	3107

1	2	4	8	16	32	64	
0x01	0x02	0x04	0x08	0x10	0x20	0x40	
20Hz	eDiff	RTK Rover, RTK Base, Raw Out	RTK Base, Raw Out	Raw Out	L2	GLONASS	Date Code (Additive Code)
		Y	Y		Y	Y	3108
Y		Y	Y		Y	Y	3109
	Y	Y	Y		Y	Y	3110
Y	Y	Y	Y		Y	Y	3111
				Y	Y	Y	3112
Y				Y	Y	Y	3113
	Y			Y	Y	Y	3114
Y	Y			Y	Y	Y	3115
		Y		Y	Y	Y	3116
Y		Y		Y	Y	Y	3117
	Y	Y		Y	Y	Y	3118
Y	Y	Y		Y	Y	Y	3119
			Y	Y	Y	Y	3120
Y			Y	Y	Y	Y	3121
	Y		Y	Y	Y	Y	3122
Y	Y		Y	Y	Y	Y	3123
		Y	Y	Y	Y	Y	3124
Y		Y	Y	Y	Y	Y	3125
	Y	Y	Y	Y	Y	Y	3126
Y	Y	Y	Y	Y	Y	Y	3127

Date Code	Hex Code	Update Rate	Subscription Description
3000	0	10Hz	L1, GPS
3001	1	20Hz	L1, GPS
3002	2	10Hz	L1, GPS, eDiff
3003	3	20Hz	L1, GPS, eDiff
3004	4	10Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3005	5	20Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3006	6	10Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3007	7	20Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3008	8	10Hz	L1, GPS, RTK Base, Raw Ou
3009	9	20Hz	L1, GPS, RTK Base, Raw Ou
3010	A	10Hz	L1, GPS, eDiff, RTK Base, Raw Ou
3011	B	20Hz	L1, GPS, eDiff, RTK Base, Raw Ou
3012	C	10Hz	L1, GPS, RTK Rover, RTK Base, Raw Out

Date Code	Hex Code	Update Rate	Subscription Description
3013	D	20Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3014	E	10Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3015	F	20Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3016	10	10Hz	L1, GPS, Raw Out
3017	11	20Hz	L1, GPS, Raw Out
3018	12	10Hz	L1, GPS, eDiff, Raw Out
3019	13	20Hz	L1, GPS, eDiff, Raw Out
3020	14	10Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3021	15	20Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3022	16	10Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3023	17	20Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3024	18	10Hz	L1, GPS, RTK Base, Raw Ou
3025	19	20Hz	L1, GPS, RTK Base, Raw Ou
3026	1A	10Hz	L1, GPS, eDiff, RTK Base, Raw Ou
3027	1B	20Hz	L1, GPS, eDiff, RTK Base, Raw Ou
3028	1C	10Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3029	1D	20Hz	L1, GPS, RTK Rover, RTK Base, Raw Out
3030	1E	10Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3031	1F	20Hz	L1, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3032	20	10Hz	L1/L2, GPS
3033	21	20Hz	L1/L2, GPS
3034	22	10Hz	L1/L2, GPS, eDiff
3035	23	20Hz	L1/L2, GPS, eDiff
3036	24	10Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3037	25	20Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3038	26	10Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3039	27	20Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3040	28	10Hz	L1/L2, GPS, RTK Base, Raw Ou
3041	29	20Hz	L1/L2, GPS, RTK Base, Raw Ou
3042	2A	10Hz	L1/L2, GPS, eDiff, RTK Base, Raw Ou
3043	2B	20Hz	L1/L2, GPS, eDiff, RTK Base, Raw Ou
3044	2C	10Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3045	2D	20Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3046	2E	10Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3047	2F	20Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3048	30	10Hz	L1/L2, GPS, Raw Out
3049	31	20Hz	L1/L2, GPS, Raw Out
3050	32	10Hz	L1/L2, GPS, eDiff, Raw Out
3051	33	20Hz	L1/L2, GPS, eDiff, Raw Out

Date Code	Hex Code	Update Rate	Subscription Description
3052	34	10Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3053	35	20Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3054	36	10Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3055	37	20Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3056	38	10Hz	L1/L2, GPS, RTK Base, Raw Ou
3057	39	20Hz	L1/L2, GPS, RTK Base, Raw Ou
3058	3A	10Hz	L1/L2, GPS, eDiff, RTK Base, Raw Ou
3059	3B	20Hz	L1/L2, GPS, eDiff, RTK Base, Raw Ou
3060	3C	10Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3061	3D	20Hz	L1/L2, GPS, RTK Rover, RTK Base, Raw Out
3062	3E	10Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3063	3F	20Hz	L1/L2, GPS, eDiff, RTK Rover, RTK Base, Raw Out
3064	40	10Hz	L1, GPS/GLONASS
3065	41	20Hz	L1, GPS/GLONASS
3066	42	10Hz	L1, GPS/GLONASS, eDiff
3067	43	20Hz	L1, GPS/GLONASS, eDiff
3068	44	10Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3069	45	20Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3070	46	10Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3071	47	20Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3072	48	10Hz	L1, GPS/GLONASS, RTK Base, Raw Ou
3073	49	20Hz	L1, GPS/GLONASS, RTK Base, Raw Ou
3074	4A	10Hz	L1, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3075	4B	20Hz	L1, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3076	4C	10Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3077	4D	20Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3078	4E	10Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3079	4F	20Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3080	50	10Hz	L1, GPS/GLONASS, Raw Out
3081	51	20Hz	L1, GPS/GLONASS, Raw Out
3082	52	10Hz	L1, GPS/GLONASS, eDiff, Raw Out
3083	53	20Hz	L1, GPS/GLONASS, eDiff, Raw Out
3084	54	10Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3085	55	20Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3086	56	10Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3087	57	20Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3088	58	10Hz	L1, GPS/GLONASS, RTK Base, Raw Ou
3089	59	20Hz	L1, GPS/GLONASS, RTK Base, Raw Ou
3090	5A	10Hz	L1, GPS/GLONASS, eDiff, RTK Base, Raw Ou

Date Code	Hex Code	Update Rate	Subscription Description
3091	5B	20Hz	L1, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3092	5C	10Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3093	5D	20Hz	L1, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3094	5E	10Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3095	5F	20Hz	L1, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3096	60	10Hz	L1/L2, GPS/GLONASS
3097	61	20Hz	L1/L2, GPS/GLONASS
3098	62	10Hz	L1/L2, GPS/GLONASS, eDiff
3099	63	20Hz	L1/L2, GPS/GLONASS, eDiff
3100	64	10Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3101	65	20Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3102	66	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3103	67	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3104	68	10Hz	L1/L2, GPS/GLONASS, RTK Base, Raw Ou
3105	69	20Hz	L1/L2, GPS/GLONASS, RTK Base, Raw Ou
3106	6A	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3107	6B	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3108	6C	10Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3109	6D	20Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3110	6E	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3111	6F	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3112	70	10Hz	L1/L2, GPS/GLONASS, Raw Out
3113	71	20Hz	L1/L2, GPS/GLONASS, Raw Out
3114	72	10Hz	L1/L2, GPS/GLONASS, eDiff, Raw Out
3115	73	20Hz	L1/L2, GPS/GLONASS, eDiff, Raw Out
3116	74	10Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3117	75	20Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3118	76	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3119	77	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3120	78	10Hz	L1/L2, GPS/GLONASS, RTK Base, Raw Ou
3121	79	20Hz	L1/L2, GPS/GLONASS, RTK Base, Raw Ou
3122	7A	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3123	7B	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Base, Raw Ou
3124	7C	10Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3125	7D	20Hz	L1/L2, GPS/GLONASS, RTK Rover, RTK Base, Raw Out
3126	7E	10Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out
3127	7F	20Hz	L1/L2, GPS/GLONASS, eDiff, RTK Rover, RTK Base, Raw Out

Determining the Receiver Type and Current Application

To determine the current receiver type, use the [JT](#) command. Table 1 shows the receiver type indicated by the JT response.

Table 1: \$JT Response and Receiver Type	
\$JT Response	Receiver Type
SX1x	SX-1
SX2x	Crescent
SLXx	SLX2/SLX3
DF2x	Eclipse
DF3x	Eclipse II
MF3x	miniEclipse

The 'x' in the responses represents the receiver's current application. For example, if x = i, as in SX2i, 'i' is the application code for e-Dif.

Table 2 shows the application for the application code in the JT response.

Table 2: \$JT Response and Application	
\$JT Responses with Application Code	Receiver Application
r	RTK rover
b	RTK base
i	e-Dif
g	OmniSTAR
g	WAAS
g	Standalone
a	Vector

Configuring the Receiver

You can configure all aspects of receiver operation through any serial port using NMEA 0183 commands. You can:

- Select one of the two on-board applications
 - Two applications may be loaded at the same time, but only one can be active
 - You can select the active application through serial commands or through menu options on products with displays
- Set the baud rate of both communication ports
- Select NMEA 0183 data messages to output on the serial ports and select the output rate of each message
- Set the maximum differential age cut-off
- Set the satellite elevation angle cut-off mask

The appropriate commands are described in [Commands and Messages](#).

Configuring the Data Message Output

In addition to its differential-only Port D, the receiver features three primary bi-directional ports referred to as A, B, and C. You can configure GPS data messages for all three ports by sending NMEA 0183 commands to the receiver module through all its communication ports. You can configure the output of Port B through A, for instance, and vice versa. The [JASC](#) NMEA message allows you to turn the messages on or off as required.

Note: For receivers that have a USB port that supports writing to a USB flash drive you can specify Port T as a port to receive messages.

In the examples below where you can specify the port, use 'PORTT' to specify Port T.

'THIS' Port and the 'OTHER' Port

The NMEA 0183 interface for Port A and B both use 'THIS' and 'OTHER' terminology.

- **THIS port**
The port you are currently connected to for inputting commands. To get the data output through THIS port it is not necessary to specify 'this' (see Example 1 below).
- **The OTHER port**
To specify the OTHER port for the data output, you need to include 'OTHER' in the command. See the two examples following which are both based on you being connected to Port B.

Example 1:

To turn the [GPGBA](#) message on at 5 Hz on Port B, use the following command:

```
$JASC,GPGBA,5<CR><LF>
```

Because B is THIS it does not have to be specified.

Example 2:

To turn the GPGBA message on at an output rate of 5 Hz on Port A, use the following command:

```
$JASC,GPGBA,5,OTHER<CR><LF>
```

Because B is THIS and A is OTHER, you have to specify OTHER. In contrast, when turning messages on or off on Port C from Port A or Port B, you must specify Port C in the command.

Example 3:

To turn the [GPGLL](#) NMEA 0183 message on at 10 Hz on Port C, use the following command:

```
$JASC,GPGLL,10,PORTC<CR><LF>
```

As with Port A and B, when communicating directly with Port C, you do not need to specify anything at the end of the message. See [Commands and Messages](#) for more information.

Saving the Receiver Configuration

Each time the configuration of the receiver is changed, the new configuration should be saved so the receiver does not have to be reconsidered for the next power cycle.

To save the settings:

- Issue the [JSAVE](#) command. The receiver records the current configuration to non-volatile memory. The receiver indicates when the save process, which takes about five seconds, is complete.

Using Port D for RTCM Input

The receiver has a port designed to accommodate externally supplied corrections input according to the RTCM SC-104 protocol. Port D provides this functionality although it has been fixed to operate at a baud rate of 9600 (8 data bits, no parity, and 1 stop bit, that is, 8-N-1).

To use Port D of the receiver for correction input, you must set the receiver to operate in beacon differential mode using the following command:

```
$JDIFF, BEACON<CR><LF>
```

This command was designed to “turn on” Port D differential operation in our products because many use the Hemisphere GPS SBX beacon module interfaced to Port D.

Note: The receiver is compatible with RTCM SC-104 message types 1-3, 5-7, 9 and 16 although not all the message types contain differential data.

To return to using SBAS as the correction source, send the following command to the receiver:

```
$JDIFF, WAAS<CR><LF>
```

See [Commands and Messages](#) for detailed information on NMEA 0183 messages supported by the receiver.

SBX-4 Database Mode

Enabling Database Mode

Database mode is automatically enabled when the SBX-4 receives a valid RMC message on Port 0. This requires the baud rate of Port 0 to be the same as the corresponding GPS receiver port.

Performance in Database Mode

In most installations Database mode will result in faster initial acquisition and better GPS accuracy compared to Auto mode.

In some installations Database mode may not work as well as Auto mode for the following reasons:

- The closest station is not in the station database and the SBX-4 has not yet received a Type7 Almanac message. Most stations now broadcast the Almanac message every ten minutes. Assuming the SBX-4 can tune to a surrounding station and receive a Type7 message, it will update the station database and automatically retune to the closest station.
- Signal quality in the area is poor. IEC61108-4 requires the receiver to switch away from a station when WER rises above 10%. For installations that do not need to comply with IEC61108-4 this threshold can be increased as usable corrections can be obtained for word error rates up to 50%.

Available Production Configuration Settings

Disable the automatic switch to Database mode:	\$PCSI , 8 , NITRAM , A
Enable weak signal tracking (WER of 50%):	\$PCSI , 8 , NITRAM , W
Enable legacy Q value output (in place of WER):	\$PCSI , 8 , NITRAM , Q
Set SBX-4 to factory defaults:	\$PCSI , 8 , NITRAM , E

PocketMAX Utility

PocketMAX Overview

PocketMax3 is a freely available utility designed for use with several Hemisphere GPS products, including:

- Crescent OEM
- Eclipse OEM
- Eclipse II OEM
- miniEclipse OEM
- Crescent Vector OEM
- Crescent Vector II OEM
- Crescent A100
- Crescent R100 Series
- Crescent XF100 Series
- Crescent V100 series
- Crescent VS100 series
- Eclipse A220 Series
- Eclipse R220
- Eclipse II R320

PocketMax3 was not designed specifically for any one product alone (it supports features not offered by every product); however, the interface may be used for all I/O operations. PocketMax3 runs on the Windows .NET framework, version 3.5 or later, allowing it to operate on several Windows platforms (Windows 2000, ME, XP, Vista, 7, Mobile, etc).

This software offers you the following flexibility:

- Tune your beacon, WAAS, OmniSTAR and GLONASS receivers and monitor reception
- Configure GPS and GNSS message output and port settings
- Configure and monitor an RTK base station
- Configure and monitor Vector related settings
- Record various types of data

The current version of PocketMax3 PC can be downloaded from the Hemisphere GPS website, or it can be made available to you by contacting Hemisphere GPS. After saving the PocketMax3 executable to your computer, start the program by double-clicking the file name or icon.

You will need to have the Windows .NET framework installed on your PC or mobile device. Follow the link for a PC install from the same webpage with the PocketMax3 download. Once you have the PocketMax3 executable appropriate for your mobile device's operating system, you can copy it over to your mobile device to whichever folder you wish. To start the program, navigate to the executable on your mobile device and tap the file.

Note: This technical reference provides summary information about what you can use PocketMax3 for. For details on how you use PocketMAX—including navigating through the menus options and tabs—refer to the PocketMAX User Guide available for download from www.hemispheregps.com.

PocketMAX Key Uses

Use PocketMAX to:

- Tune your GPS, Beacon, SBAS and OmniSTAR receiver
- Monitor GPS, Beacon, SBAS, and OmniSTAR reception
- Configure GPS message output and port settings
- Configure and monitor Vector-related settings
- Record various types of data

Because PocketMAX PC and PocketMAX were not designed specifically for one receiver, they support features not offered by some receivers, such as tracking of the OmniSTAR differential service and display of our Vector product's true heading. However, the interface may be used for all I/O operations.

PocketMAX PC runs on any PC with Windows 95, 98, or NT 4.0+ (Windows 2000 and Windows XP). Screen resolution of 800x600 or greater is recommended. One of the receiver's serial ports must be connected to a COM port on the computer.

You can download the current version of PocketMAX PC, or PocketMAX, from www.hemispheregps.com.

PocketMAX Startup

When you start PocketMAX you'll first briefly see the Welcome screen then the (untitled) startup configuration screen. Both are shown below.



Use the startup configuration screen to specify the COM port and baud rate of the receiver.

PocketMAX Features

The following tables summarize the screen content for the menu options and their respective tabs.

- [GPS tabs](#)
- [Differential Source tabs](#)
 - SBAS
 - BEAC
 - LBND (OmniSTAR)
 - e-Dif
 - L-Dif BASE
- [TMNL \(Terminal\) tabs](#)
- [LOGS tabs](#)
- [HDG \(Heading\) tabs](#)

PocketMAX GPS Tabs

Tab	Description of Content/Use
Pos'n (Position)	All the main position information including latitude and longitude, altitude, speed and precision. You can select a differential source from within this tab as well as through the differential source menu.
Sats (Satellite)	Provides a sky plot of viewable satellites, how many satellites the receiver is tracking, the PRN numbers of the satellites are being tracked and the BER (Bit Error Rate) of the differential source.
Setup	Change the configuration of the receiver including turning NMEA messages on or off, the elevation mask, the maximum COAST age and the baud rates.
Precision	Provides a graphical representation of horizontal accuracy in the form of an error ellipse. It also provides configurable numerical precision in northing, easting, and altitude components.
Plot	Plots the northing and easting error over time and enables you to adjust the scale and timeline. You can monitor performance over a time period with respect to either a known coordinate or an arbitrary one.
About	Provides current firmware information.

Differential Source Tabs

Differential source tabs can be any of the following:

SBAS Tabs

Tab	Description of Content/Use
Status	Provides details of the satellites being used in the SBAS differential system, which covers both WAAS and EGNOS. The PRNs, longitudes, elevation, azimuth, and the BER of the satellites being tracked are also shown.
Plot	Charts and gives a bar graph of the BER of up to two SBAS Satellites being tracked.

BEAC Tabs

Tab	Description of Content/Use
Status	Provides details of the beacon station providing corrections, including the name (if known), the frequency, the MSK rate, and the SS and SNR values.
Tune	Gives you the option to automatically tune to the strongest signal, specifying a frequency or MSK bit rate, or selecting a station by region.
Plot	Charts the signal strength, the SNR or the frequency of the beacon signal.

LBND (OmniSTAR) Tabs

Tab	Description of Content/Use
Status	Provides the name (if available), the frequency and data rate of the L-Band satellite currently being used. Also displays the BER, the location and status of the satellite.
Tune	Provides the name (if available), the frequency and data rate of the L-Band satellite currently being used and provides the option of tuning manually by frequency and data rate, automatically or by the name of the satellite.
Subscription	Provides the begin and expiration dates of the subscription as well as the serial number of the unit and the countdown timer (gives you the amount of time you have left for your subscription).

e-Dif Tabs

Tab	Description of Content/Use
Setup	Provides options to configure the receiver for e-Dif operation. For receivers with valid subscriptions, this screen enables you to initialize based on a multiple run - unknown control point; multiple run - known reference point (entered as the latitude, longitude and height of a reference position); or a single run.

L-Dif BASE Tabs

Tab	Description of Content/Use
Setup	Provides options to set the latitude, longitude and height of a reference position and to select the port the receiver uses to connect to a radio that broadcasts local differential corrections.

TMNL Tabs

Tab	Description of Content/Use
Terminal	Provides direct terminal access to the receiver for issuing commands and observing responses.
Hot Keys	Enables you to set up frequently used commands and assign them to the buttons in the Terminal tab. There are four levels of hot key, each with nine buttons giving you thirty-six shortcut keys for issuing commands.

LOGS Tabs

Tab	Description of Content/Use
NMEA	Enables you to set up NMEA messages to be logged.
Raw Data	Enables you to log the raw binary Bin95 and Bin96 messages for post-processing.
Binary	Enables you to log a variety of binary messages.
Points	Enables you to log a point each time you press the Log Point button.
Polygon	Enables you to log polygons and displays the enclosed area.

HDG Tabs

Tab	Description of Content/Use
Status	Enables you to set up NMEA messages to be logged.
Setup	Enables you to log the raw binary Bin95 and Bin96 messages for post-processing.
Plot	Enables you to log a variety of binary messages.

Commands and Messages

The receiver supports a selection of NMEA 0183 messages, proprietary messages that conform to NMEA 0183 standards, and Hemisphere GPS proprietary binary messages. It is your decision as a systems designer whether or not to support a NMEA 0183-only software interface or a selection of both NMEA 0183 and binary messages.

All Crescent and Eclipse receivers are configured with NMEA 0183 commands and can output NMEA 0183 messages. In addition to NMEA 0183, some receivers can be configured using NMEA 2000 commands and can output NMEA 2000 messages.

Commands

- [General operation and configuration commands](#)
- [GPS commands](#)
- [SBAS commands](#)
- [e-Dif commands](#)
- [Crescent Vector commands and messages](#)
- [GLONASS commands and messages](#)
- [DGPS base station commands](#)
- [Local differential and RTK commands](#)
- [Beacon receiver commands and messages](#)
- [NMEA 0183 SBX queries](#)
- [OmniSTAR commands](#)
- [RAIM commands](#)

Messages

- [Data messages](#)
- [Binary messages](#)
- [NMEA 2000 CAN messages](#)

NMEA 0183 Message Format

NMEA 0183 messages (sentences) have the following format:

`$XXYYY,ZZZ,ZZZ,ZZZ...*CC<CR><LF>`

where:

Element	Description
\$	Message header character
XX	NMEA 0183 talker field (GP indicates a GPS talker)
YYY	Type of GPS NMEA 0183 message
ZZZ	Variable length message fields
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Null (empty) fields occur when there is no information for that field. You can use the [JNP](#) command to specify the number of decimal places output in the [GPGGA](#) and [GPGLL](#) messages.

What does <CR><LF> mean?

The literal translation means "Carriage Return, Line Feed." They are terms used in computer programming languages to describe the end of a line or string of text. If you are writing your own communication software for a receiver, see some of the examples below. If you are already using a program such as PocketMAX, when you click to send a command to the receiver, the program takes care of adding the carriage return and line feed to the end of the text string for you. If you are using HyperTerminal or other terminal software, typically the Enter key on your keyboard is set to send the <CR><LF> pair. You may need to define this in the setup section of the terminal software. Some software may treat the Enter key on your numeric keypad differently than the main Enter key in the main QWERTY section of the keyboard – use the main Enter key for best results.

Originally, the carriage return and line feed characters were for use with printers. The carriage return character would signal the printer to send the print head back to the left edge of the page on the current line of text. The line feed command instructed the printer to advance the paper one line. Today, electronics often use the carriage return and line feed instructions to signify the end of a string of text, prompting the device to process the string and execute the instructions sent in the text string.

Electronics use different ways to represent the <CR><LF> characters. In ASCII numbers, <CR> is represented as 13 in decimal, or 0D in hexadecimal. ASCII for <LF> is 10 decimal, or 0A hexadecimal. Some computer languages use different ways to represent <CR><LF>. Unix and C language can use "\x0D\x0A". C language can also use "\r\n" in some instances. Java may use CR+LF. In Unicode, carriage return is U+000D, and line feed is U+000A. It is advised to clearly understand how to send these characters if you are writing your own interface software.

Command/Query/Message Types

General Operation and Configuration Commands

The following table lists the commands related to the general operation and configuration of the receiver.

Command	Description
JAIR	Specify how the receiver will respond to the dynamics associated with airborne applications
JALT	Turn altitude aiding for the receiver on or off
JAPP	Specify or query receiver application firmware
JASC,D1	Set the RD1 diagnostic information message from the receiver to on or off
JASC,VIRTUAL	Configure the receiver to have RTCM data input on one port and output through the other (when using an external correction source)
JBAUD	Specify the baud rates of the receiver or query the current setting
JBIN	Enable the output of the various binary messages supported by the receiver
JCONN	Create a virtual circuit between the A and B ports to enable communication through the receiver to the device on the opposite port
JDIFF	Specify or query the differential mode of the receiver
JDIFFX,EXCLUDE	Specify the differential sources to be excluded from operating in a multi-diff application
JDIFFX,GNSSOUT	Specify GNSS output in correction formats or query the current setting
JDIFFX,INCLUDE	Specify the differential sources to be allowed to operate in a multi-diff application
JDIFFX,SOURCE	Query the receiver for the differential source
JDIFFX,TYPE	Query the receiver for the differential type
JFLASH,DIR	Display the files on a USB flash drive
JFLASH,FILE,CLOSE	Close an open file on a USB flash drive
JFLASH,FILE,NAME	Open a specific file, append to a specific file, or display the file name of the open file on a USB flash drive
JFLASH,FILE,OPEN	Create and open a file with an automatically generated file name on a USB flash drive
JFLASH,FREESPACE	Display the free space in kilobytes (KB) on a USB flash drive
JFLASH,NOTIFY,CONNECT	Enable/disable the automatic response when a USB flash drive is inserted or removed (if port is not specified the response will be sent to the port that issued the command)
JFLASH,QUERYCONNECT	Manually verify if a USB flash drive is connected or disconnected
JI	Display receiver information, such as its serial number and firmware version
JK	Subscribe the receiver to various options, such as higher update rates, e-Dif (or base station capability) or L-Dif; or query for the current subscription expiration date when running OmniSTAR application or the receiver subscription code when running all other applications
JLIMIT	Set the threshold of estimated horizontal performance for which the DGPS position LED is illuminated or query the current setting
JMODE	Query receiver for status of JMODE settings
JMODE,FOREST	Turn the higher gain functionality (for tracking under canopy) on/off or query the current setting

Command	Description
<u>JMODE,GPSONLY</u>	Set the receiver to use GPS data in the solution or query the current setting (if GLONASS is available, setting to YES will cause the receiver to only use GPS data)
<u>JMODE,L1ONLY</u>	Set the receiver to use L1 data even if L2 data is available or query the current setting
<u>JMODE,MIXED</u>	Include satellites that do not have differential corrections in the solution
<u>JMODE,NULLNMEA</u>	Enable/disable output of NULL fields in NMEA 0183 messages when there is no fix (when position is lost)
<u>JMODE,SBASR</u>	Enable/disable SBAS ranging
<u>JMODE,TIMEKEEP</u>	Enable/disable continuous time updating in NMEA 0183 messages when there is no fix (when position is lost)
<u>JMODE,TUNNEL</u>	Enable/disable faster reacquisition after coming out of a tunnel or query the current setting
<u>JPOS</u>	Speed up the initial acquisition when changing continents with the receiver or query the receiver for the current position of the receiver
<u>JQUERY,GUIDE</u>	Query the receiver for its determination on whether or not it is providing suitable accuracy after both the SBAS and GPS have been acquired (up to five minutes)
<u>JRELAY</u>	Send user-defined text out of a serial port
<u>JRESET</u>	Reset the receiver to its default operating parameters by turning off outputs on all ports, saving the configuration, and setting the configuration to its defaults
<u>JSAVE</u>	Send this command after making changes to the operating mode of the receiver
<u>JSHOW</u>	Query the current operating configuration of the receiver
<u>JT</u>	Query the receiver for its GPS engine type

Note: Use the [JSAVE](#) command to save changes you need to keep and wait for the \$>SAVE COMPLETE response.

GPS Commands

The following table lists the commands supported by the internal GPS engine for its configuration and operation.

Command	Description
JAGE	Specify maximum DGPS (COAST) correction age (6 to 8100 seconds)
JASC,GP	Enable the GPS data messages at a particular update rate to be turned on or off
JMASK	Specify the elevation cutoff mask angle for the GPS engine
JNP	Specify the number of decimal places output in the GPGGA and GPGLL messages
JOFF	Turn off all data messages being output through the current port or other port
JOFF,ALL	Turn off all data messages being output through all ports
JSMOOTH	Set the carrier smoothing interval (15 to 6000 seconds) or query the current setting
JTAU,COG	Set the course over ground (COG) time constant (0.00 to 3600.00 seconds) or query the current setting
JTAU,SPEED	Set the speed time constant (0.00 to 3600.00 seconds) or query the current setting

Note: Use the [JSAVE](#) command to save changes you need to keep and wait for the \$>SAVE COMPLETE response.

SBAS Commands

The following table lists the commands supported by the SBAS demodulator for its control and operation.

Command	Description
JASC,D1	Set the RD1 diagnostic information message from the receiver to on or off
JASC,RTCM	Configure the receiver to output RTCM version 2 DGPS corrections from SBAS or beacon through either receiver serial port
JGEO	Display information related to the current frequency of SBAS and its location in relation to the receiver's antenna
JWAASPRN	Change the SBAS PRNs in memory or query the receiver for current PRNs in memory

Note: Use the [JSAVE](#) command to save changes you need to keep and wait for the \$>SAVE COMPLETE response.

e-Dif Commands

The following table lists the commands supported by the e-Dif application for its control and operation.

Command	Description
JRAD,1	Display the current reference position in e-Dif applications only
JRAD,1,LAT,LON,HEIGHT	Use this command—a derivative of the JRAD,1,P command—when absolute positioning is required in e-Dif applications only
JRAD,1,P	<p>e-Dif: Record the current position as the reference with which to compute e-Dif corrections. This would be used in relative mode as no absolute point information is specified.</p> <p>DGPS Base Station: Record the current position as the reference with which to compute Base Station corrections in e-Dif applications only. This would be used in relative mode as no absolute point information is specified</p>

Command	Description
JRAD.2	Forces the receiver to use the new reference point (you normally use this command following a JRAD.1 type command)
JRAD.3	Invoke the e-Dif function once the unit has started up with the e-Dif application active, or, update the e-Dif solution (calibration) using the current position as opposed to the reference position used by the JRAD.2 command
JRAD.7	Turn auto recalibration on or off

Note: Use the [JSAVE](#) command to save changes you need to keep and wait for the \$>SAVE COMPLETE response.

Crescent Vector Commands and Messages

The following table lists the commands related to the GPS heading aspect of the Crescent Vector OEM heading system.

Command	Description
JASC	Turn on different messages
JATT,COGTAU	Set the course over ground (COG) time constant (0.0 to 3600.0 seconds) or query the current setting
JATT,CSEP	Query for the current separation between GPS antennas
JATT,EXACT	Enable/disable internal filter reliance on the entered antenna separation or query the current setting
JATT,FLIPBRD	Turn the flip feature on/off (allowing you to install the Crescent Vector board upside down) or query the current feature status
JATT,GYROAID	Turn gyro aiding on or off or query the current setting
JATT,HBIAS	Set the heading bias or query the current setting
JATT,HELP	Show the available commands for GPS heading operation and status
JATT,HIGHMP	Set/query the high multipath setting for use in poor GPS environments
JATT,HRTAU	Set the heading rate time constant or query the current setting
JATT,HTAU	Set the heading time constant or query the current setting
JATT,LEVEL	Turn level operation on or off or query the current setting
JATT,MSEP	Manually set the GPS antenna separation or query the current setting
JATT,NEGTILT	Turn the negative tilt feature on or off or query the current setting
JATT,NMEAHE	Instruct the Crescent Vector to preface the HDG, HDM, HDT, and ROT messages with GP or HE
JATT,PBIAS	Set the pitch/roll bias or query the current setting
JATT,PTAU	Set the pitch time constant or query the current setting
JATT,ROLL	Configure the Crescent Vector for roll or pitch GPS antenna orientation
JATT,SEARCH	Force the Crescent Vector to reject the current GPS heading solution and begin a new search
JATT,SPDTAU	Set the speed time constant (0.0 to 3600.0 seconds) or query the current setting
JATT,SUMMARY	Display a summary of the current Crescent Vector settings
JATT,TILTAID	Turn tilt aiding on or off or query the current setting
JATT,TILTCAL	Calibrate tilt aiding or query the current feature status

The following table lists Crescent Vector messages.

Message	Description
GPDTM	Datum reference
GPGGA	GPS fix data
GPGLL	Geographic position - latitude/longitude
GPGNS	GNSS fix data
GPGRS	GNSS range residuals
GPGSA	GNSS DOP and active satellites
GPGST	GNSS pseudorange error statistics
GPGSV	GNSS satellite in view
GPHDG/HEHDG	Provide magnetic deviation and variation for calculating magnetic or true heading
GPHDM/HEHDM	Provide magnetic heading of the vessel derived from the true heading calculated
GPHDT/HEHDT	Provide true heading of the vessel
GPHEV	Heave value in meters
GPRMC	Recommended minimum specific GNSS data
GPROT/HEROT	Contains the vessel's rate of turn (ROT) information
GPRRE	Range residual message
GPVTG	Course over ground and ground speed
GPZDA	Time and date
PASHR	Time, heading, roll, and pitch data in one message
PSAT_GBS	Satellite fault detection used for RAIM
PSAT_HPR	Proprietary NMEA sentence that provides the heading, pitch/roll information and time in a single message
PSAT_INTLT	Proprietary NMEA sentence that provides the title measurement from the internal inclinometer (in degrees)
TSS1	Heading, pitch, roll, and heave message in the commonly used TSS1 message format

GLONASS Commands and Messages

The following table lists the commands applicable to GLONASS-capable receivers.

Command	Description
JASC, GL	Enable the GLONASS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.
JMODE, GPSONLY	Set the receiver to use GPS data in the solution or query the current setting (if GLONASS is available, setting to YES will cause the receiver to only use GPS data)
JNMEA, GGA, ALLGNSS	Configure the GGA string to include full GNSS information (the number of used GLONASS satellites will be included in the GPGGA message) or query the current setting

The following table lists the messages applicable to GLONASS-capable receivers.

Message	Description
Bin62	GLONASS almanac information
Bin65	GLONASS ephemeris information
Bin66	GLONASS L1 code and carrier phase information
Bin69	GLONASS L1 diagnostic information
GLMLA	GLONASS almanac data - contains complete almanac data for one GLONASS satellite (multiple sentences may be transmitted, one for each satellite in the GLONASS constellation)

DGPS Base Station Commands

The following table lists the commands supported by the base station feature for its control and operation.

Command	Description
JRAD,1	Display the current reference position in e-Dif applications only
JRAD,1,LAT,LON,HEIGHT	Use this command—a derivative of the JRAD,1,P command—when absolute positioning is required in e-Dif applications only
JRAD,1,P	e-Dif: Record the current position as the reference with which to compute e-Dif corrections. This would be used in relative mode as no absolute point information is specified. DGPS Base Station: Record the current position as the reference with which to compute Base Station corrections in e-Dif applications only. This would be used in relative mode as no absolute point information is specified
JRAD,9,1,1	Initialize the Base Station feature and use the previously entered point, either with \$JRAD,1,P or \$JRAD,1,LAT,LON,HEIGHT , as the reference with which to compute Base Station corrections in e-Dif applications only. Use this for both relative mode and absolute mode.

Local Differential and RTK Commands and Messages

The following table lists the commands supported by Local Differential ([L-Dif](#)) and [RTK](#) feature for its control and operation.

Command	Description
JASC,CMR	Set the proprietary CMR messages to on or off to provide corrections to the rover (only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode)
JASC,DFX	Set the proprietary DFX messages to on or off to provide corrections to the rover (only applies to a Crescent base receiver when using L-Dif or RTK mode)
JASC,ROX	Set the proprietary ROX messages to on or off to provide corrections to the rover (only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode)
JASC,RTCM3	Set the RTCM version 3 messages to on or off to provide corrections to the rover (only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode)
JASC,PSAT,RTKSTAT	Configure the receiver to output the most relevant parameters affecting RTK
JQUERY,RTKSTAT	Perform a one-time query of the most relevant parameters that affect RTK
JRTK,1	Show the receiver's reference position (can issue command to base station or rover)
JRTK,1,LAT,LON,HEIGHT	Set the receiver's reference position to the coordinates you enter (can issue command to base station or rover)
JRTK,1,P	Set the receiver's reference coordinates to the current calculated position if you do not have known coordinates for your antenna location (can issue command to base station or rover)
JRTK,5	Show the base station's transmission status for RTK applications (can issue command to base station)
JRTK,5,Transmit	Suspend or resume the transmission of RTK (can issue command to base station)
JRTK,6	Display the progress of the base station (can issue command to base station)
JRTK,12	Disable or enable the receiver to go into fixed integer mode (RTK) vs. float mode (L-Dif) - can issue command to rover

Command	Description
JRTK,17	Display the transmitted latitude, longitude, and height of the base station (can issue command to base station or rover)
JRTK,18	Display the distance from the rover to the base station, in meters (can issue command to rover)
JRTK,28	Set the base station ID transmitted in ROX/DFX/CMR/RTCM3 messages (can issue command to base station)

The following table lists the Local Differential ([L-Dif](#)) and [RTK](#) messages.

Message	Description
PSAT,RTKSTAT	Contains the most relevant parameters affecting RTK

Beacon Receiver Commands and Messages

If integrating a Hemisphere GPS SBX beacon module with the receiver GPS engine, Hemisphere GPS recommends interfacing the beacon receiver to Port D of the receiver engine. Hemisphere GPS has implemented some command and message pass-through intelligence for such an integration. In this configuration you can issue the commands in the following table to the beacon receiver through either Port A, Port B, or Port C of the receiver.

The following table lists the beacon receiver commands found in this Help file.

Command	Description
GPMSK	Tune beacon the receiver and turn on diagnostic information
PCSI,1,1	Obtain beacon status information from the SBX beacon engine inside the receiver
PCSI,3,2	Display the ten closest beacon stations
PCSI,3,3	Display the contents of the beacon station database

The following table lists the beacon receiver messages found in this Help file.

Message	Description
CRMSK	Operational status message of SBX
CRMSS	Performance status message of SBX

NMEA 0183 SBX Queries

The following table lists the standard and Hemisphere GPS proprietary NMEA 0183 queries accepted by the SBX.

When you issue these queries to the SBX primary communications port, the response messages are output interspersed with RTCM correction information. This may cause conflicts with a GPS receiver's ability to compute differential corrected solutions. By sending these queries to the SBX secondary communications port the flow of RTCM corrections on the primary port will not be interrupted.

Query	NMEA 0183 Query Type	Description
GPCRQ_MSK	Standard	Query the SBX for its operational status
GPCRQ_MSS	Standard	Query the SBX for its performance status
PCSI_0	Hemisphere GPS proprietary	Query the SBX to output a list of available proprietary PCSI commands
PCSI_1	Hemisphere GPS proprietary	Query the SBX for a selection of parameters related to the operational status of its primary channel
PCSI_2	Hemisphere GPS proprietary	Query the SBX to output a selection of parameters related to the operational status of its secondary channel
PCSI_3,1	Hemisphere GPS proprietary	Query the SBX to output the search information used for beacon selection in Automatic Beacon Search mode. The output has three frequencies per line.

OmniSTAR Commands

The following tables lists the commands accepted by the LX-1 OmniSTAR receiver to configure and monitor the OmniSTAR functionality of the receiver.

Command	Description
JBOOT,OMNI	Power down the OmniSTAR portion of the Eclipse engine and power it back up
JFREQ	Tune the OmniSTAR receiver (manually or automatically) or query the receiver for the current setting
JHP,LIMIT	Specify the OmniSTAR HP/XP convergence threshold (range is 0.0 to 1.0 m) or display the threshold as compared to the RMS value in the GPGST message
JHP,MODE,AUTOSEED	Enable or disable the AUTOSEED feature when operating in LBAND mode and using OmniSTAR XP/HP service, or query the current setting
JHP,MODE,IGNORECONVERGE	If using the JHP,LIMIT command to specify the OmniSTAR HP/XP convergence threshold, use this command to set the receiver to ignore when the OmniSTAR engine indicates it is converged or query the current setting
JHP,POS	Query the receiver for the stored position with standard deviations (StDevs) to be used with the JHP,SEED command
JHP,POS,LAT,LON,HGT	Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the JHP,SEED command
JHP,POS,LAT,LON,HGT,...,OTHER	(For use with AUTOSEED feature) Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the JHP,MODE,AUTOSEED command
JHP,POS,OTHER	(For use with AUTOSEED feature) Query the receiver for the stored position with standard deviations (StDevs) to be used with the JHP,MODE,AUTOSEED command

Command	Description
JHP,POS,PRESENT	Save the current location (lat, lon, height) and standard deviations into non-volatile memory (provided the sum of the location standard deviations (StDev) < 0.6 m) to be used with the JHP,SEED command
JHP,RESET,ACCURACY	Reset the HP convergence by forcing the solution to the current location but with very large standard deviations
JHP,RESET,ENGINE	Reset the HP engine, forcing the solution to converge (this will also force an AUTOSEED location to reconverge)
JHP,SEED	Initialize the OmniSTAR HP algorithm with the saved position and saved standard deviations
JHP,SEED,LAT,LON,HGT	Initialize the OmniSTAR HP algorithm with the given coordinates and optional standard deviations (this command has the combined effect of the JHP,POS,LAT,LON,HGT and JHP,SEED commands)
JHP,STATIC	Place the OmniSTAR HP engine into or out of static mode, or query the current setting
JHP,STATUS,AUTOSEED	Displays the status of the AUTOSEED initialization progress
JLBEAM	Display the information of each spot beam currently in use by the OmniSTAR receiver
JLXBEAM	Display spot beam debug information
JOMS	Request the raw OmniSTAR subscription information

Note: Use the JSAVE command to save changes you need to keep and wait for the \$J>SAVE COMPLETE response.

OmniSTAR HP

For Eclipse receivers you can reduce OmniSTAR HP initialization time by supplying the known position. If you know the current position coordinates accurately, the OmniSTAR algorithm can be sent with the known coordinates.

Warning! The coordinates should be known to within 2 cm (1 in) before attempting to seed the position. Any errors entered here will effect the future accuracy of the position solution.

You can query and store the current position with the following commands:

- `$JHP, POS, PRESENT`
Save the current location and standard deviations of location into memory. If the current latitude, longitude, and altitude standard deviations are cumulatively greater than 0.6 m, the current position is not stable and the command is ignored. Under this condition, the system responds with the following message:

`Present Location Not Stable`
- `$JHP, POS`
Query the receiver for the saved position and saved standard deviation
- `$JHP, POS, LAT, LON, HEIGHT`
Save the longitude, and height and optionally save related standard deviations (LatStDev, LonStDev, HgtStDev) where LAT and LON are in degrees and HEIGHT is in meters

To speed up initialization, you can seed the OmniSTAR algorithm with a position with the following command:

- `$JHP, SEED, LAT, LON, HEIGHT`
Where LAT and LON are in degrees and HEIGHT is in meters. When the current receiver position is greater than 12 m (in the horizontal plane) from the seed position, the receiver responds with the following message and aborts the command:

`Current Position Too Far From Seed`

RAIM Commands

RAIM (Receiver Autonomous Integrity Monitoring) is a GPS integrity monitoring scheme that uses redundant ranging signals to detect a satellite malfunction resulting in a large range error. The Hemisphere GPS products use RAIM to alert users when errors have exceeded a user-specified tolerance. RAIM is available for SBAS, Beacon, and OmniSTAR applications.

The following table lists the available RAIM commands.

Command	Description
JRAIM	Specify the parameters of the RAIM scheme that affect the output of the PSAT_GBS message or query the current setting

Data Messages

Note: 20 Hz output is only available with a 20 Hz subscription.

Message	Maximum Rate	Description
GPALM	1 Hz	GPS almanac data
GPDTM	1 Hz	Datum reference
GPGGA	20 Hz	Detailed GPS position information
GPGLL	20 Hz	Latitude and longitude data
GPGNS	20 Hz	Fixes data for single or combined satellite navigation systems
GPGRS	20 Hz	Supports Receiver Autonomous Integrity Monitoring (RAIM)
GPGSA	1 Hz	GPS DOP and active satellite information
GPGST	1 Hz	GNSS pseudorange error statistics
GPGSV	1 Hz	GNSS satellite in view
GPHDG/HEHDG	20 Hz	Magnetic deviation and variation for calculating magnetic or true heading
GPHDM/HEHDM	20 Hz	Magnetic heading of the vessel derived from the true heading calculated
GPHDT/HEHDT	20 Hz	True heading of the vessel
GPHEV	20 Hz	Heave value in meters
GPRMC	20 Hz	Recommended minimum specific GNSS data
GPROT/HEROT	20 Hz	Vessel's rate of turn (ROT) information
GPRRE	1 Hz	Range residual message
GPVTG	20 Hz	Course over ground and ground speed
GPZDA	20 Hz	UTC time and date information
PSAT_GBS	1 Hz	Used to support Receiver Autonomous Integrity Monitoring (RAIM)
PSAT_HPR	20 Hz	Proprietary NMEA message that provides the heading, pitch, roll, and time in a single message
PSAT_INTLT	1 Hz	Proprietary NMEA message that provides the tilt measurements from the internal inclinometers (in degrees)
RD1	1 Hz	SBAS diagnostic information

Binary Messages

Message Structure

The binary messages supported by the receiver are in an Intel Little Endian format for direct read in a PC environment. More information on this format at the following web site:

<http://www.cs.umass.edu/~verts/cs32/endianness.html>

Each binary message begins with an 8-byte header and ends with a carriage return, line feed pair (0x0D, 0x0A). The first four characters of the header is the ASCII sequence \$BIN.

The following table provides the general binary message structure.

Component	Description	Type	Bytes	Values
Header	Synchronization String	4 byte string	4	\$BIN
	Block ID - type of binary message	Unsigned short	2	1, 2, 80, 93, 94, 95, 96, 97, 98, or 99
	DataLength - the length of the binary messages	Unsigned short	2	52, 16, 40, 56, 96, 128, 300, 28, 68, or 304
Data	Binary Data - varying fields of data with a total length of DataLength bytes	Mixed fields	52, 16, 40, 56, 96, 128, 300, 28, 68, or 304	Varies - see message tables
Epilogue	Checksum - sum of all bytes of the data (all DataLength bytes); the sum is placed in a 2-byte integer	Unsigned short	2	Sum of data bytes
	CR- Carriage return	Byte	1	0D hex
	LF - Line feed	Byte	1	0A hex

Messages

Message	Description
Bin1	GPS position message (position and velocity data)
Bin2	GPS DOPs (Dilution of Precision)
Bin62	GLONASS almanac information
Bin65	GLONASS ephemeris information
Bin66	GLONASS L1 code and carrier phase information
Bin69	GLONASS L1 diagnostic information
Bin76	GPS L1/L2 code and carrier phase information
Bin80	SBAS data frame information
Bin89	SBAS satellite tracking information
Bin93	SBAS ephemeris information
Bin94	Ionospheric and UTC conversion parameters
Bin95	GPS ephemeris information
Bin96	GPS L1 code and carrier phase information
Bin97	Processor statistics

Message	Description
Bin98	Satellite and almanac information
Bin99	GPS L1 diagnostic information

NMEA 2000 CAN Messages

Message	Description
GNSSPositionData	Detailed GPS position information
GNSSPositionRapidUpdates	Abbreviated GPS position information
NMEACogSogData	GPS speed and direction information

Commands (All)

GPCRQ,MSK Command

Command Type [NMEA 0183 SBX](#)

Description Standard NMEA 0183 query to prompt the SBX for its operational status (response is the [CRMSK message](#))

You can issue this command through the secondary serial port with a standard response issued to the same port. This will not affect the output of RTCM data from the main serial port when the receiver has acquired a lock on a beacon station.

Command Format \$GPCRQ , MSK<CR><LF>

Receiver Response \$CRMSK , fff.f , X , ddd , Y , n*CC<CR><LF>

where

Response Component	Description
fff.f	Frequency in kHz (283.5 to 325)
X	Tune mode (M = manual, A = automatic, D = database)
ddd	MSK bit rate (100 or 200 bps)
Y	MSK rate selection mode (M = manual, A = automatic, D = database)
n	Period of output of CRMSS performance status message (0 to 100 seconds)

Example Response example:

\$CRMSK , 322.0 , M , 100 , A , 2*CC

The frequency is 322.0 kHz, tune mode is Manual, MSK bit rate is 100 bps, MSK rate selection mode is Automatic, and the message is output every 2 seconds.

Additional Information

GPCRQ,MSS Command

Command Type [NMEA 0183 SBX](#)

Description Standard NMEA 0183 query to prompt the SBX for its performance status (response is the [CRMSS message](#))

You can issue this command through the secondary serial port with a standard response issued to the same port. This will not affect the output of RTCM data from the main serial port when the receiver has acquired a lock on a beacon station.

Command Format \$GPCRQ,MSS<CR><LF>

Receiver Response \$CRMSS,xx,yy,fff.f,ddd*CC<CR><LF>

where

Response Component	Description
xx	Signal strength in dB μ V/m
yy	Signal-to-noise ratio (SNR) in dB
fff.f	Frequency in kHz (283.5 to 325)
ddd	MSK bit rate in bps (100 or 200)

Example Response example:

\$CRMSS,65,36,322.0,100*CC

The signal strength is 65 dB μ V/m, SNR is 36 dB, frequency is 322.0 kHz, and MSK bit rate is 100 bps.

Additional Information

GPMSK Command

Command Type [Beacon Receiver](#)

Description Beacon Tune command

Instruct the SBX to tune to a specified frequency and automatically select the correct MSK rate. When you send this command through Port A, Port B, or Port C, it is automatically routed to Port D. The resulting confirmation of this message is returned to the same port from which you sent the command.

Command Format \$GPMSK,fff.f,F,mmm,M[,n]<CR><LF>

where:

Command/Response Component	Description
fff.f	Beacon frequency in kHz (283.5 to 325) This may be left blank if the following field 'F' is set to 'A' (automatic) or 'D' (database)
F	Frequency selection mode (M = manual, A = automatic, D = database)
mmm	MSK bit rate This may be left blank if the following field 'M' is set to 'A' (automatic) or 'D' (database)
M	MSK rate selection mode (M = manual, A = automatic, D = database)
n	Period of output of CRMSS performance status message (0 to 100 seconds), where leaving the field blank will output the message once Note: This field is optional when using database tuning mode or automatic tuning mode.

Receiver Response \$CRMSS,xx,yy,fff.f,ddd*CC<CR><LF>

where

Response Component	Description
xx	Signal strength in dBμV/m
yy	Signal-to-noise ratio (SNR) in dB
fff.f	Frequency in kHz (283.5 to 325)
ddd	MSK bit rate in bps (100 or 200)

Example To instruct the SBX to tune to 310.5 kHz with a bit rate of 100 and output the CRMSS message every 20 seconds issue the following command:

```
$GPMSK,310.5,M,100,M,20<CR><LF>
```

...and the receiver response is:

```
$CRMSS,65,36,310.5,100*CC  
(repeating every n=20 seconds)
```

If using database tuning mode issue the following command:

```
$GPMSK,,D,,D<CR><LF>
```

If using automatic tuning mode issue the following command:

```
$GPMSK,,A,,A<CR><LF>
```

**Additional
Information**

When the SBX acknowledges this message, it immediately tunes to the specified frequency and demodulates at the specified rate.

When you set 'n' to a non-zero value, the SBX outputs the CRMSS message at that period through the serial port from which the SBX was tuned. When you issue this command with a non-zero 'n' value through Port B, the periodic output of the CRMSS performance status message does not impact the output of RTCM on Port A. However, when tuning the SBX with a non-zero 'n' value through Port A, the CRMSS message is interspersed with the RTCM data. Most GPS engines will not be able to filter the CRMSS message, causing the overall data to fail parity checking. When power to the SBX is removed and reapplied, the status output interval resets to zero (no output).

When tuning the SBX engine, if the 'n' field in this message is non-zero, the CRMSS message output by the SBX may interrupt the flow of RTCM data to the GPS receiver. Repower the SBX to stop the output of the CRMSS message or retune the Beacon receiver with 'n' set to zero.

JAGE Command

Command Type [GPS](#)

Description Specify maximum DGPS (COAST) correction age (6 to 8100 seconds). Using COAST technology, the receiver can use old correction data for extended periods of time.

The default setting for the receiver is 2700 seconds.

If you select a maximum correction age older than 1800 seconds (30 minutes), test the receiver to ensure the new setting meets the requirements, as accuracy will slowly drift with increasing time.

Command Format \$JAGE,AGE<CR><LF>

where 'AGE' is the maximum differential age timeout

Receiver Response \$>

Example To set the DGPS correction age to 60 seconds issue the following command:

\$JAGE,60<CR><LF>

Additional Information To query the receiver for the current DGPS correction age, issue the [JSHOW](#) command.

[What does <CR><LF> mean?](#)

JAIR Command

Command Type [General Operation and Configuration](#)

Description Specify how the receiver will respond to the dynamics associated with airborne applications or query the current setting

Command Format Specify how the receiver responds

\$JAIR, R<CR><LF>

where 'R' is the AIR mode:

- NORM - normal track and nav filter bandwidth
- HIGH - highest track and nav filter bandwidth (receiver is optimized for the high dynamic environment associated with airborne platforms)
- LOW - lowest track and nav filter bandwidth
- AUTO - default track and nav filter bandwidth, similar to NORM but automatically goes to HIGH above 30 m/sec

Query the current setting

\$JAIR<CR><LF>

Receiver Response Receiver response when specifying how the receiver responds or querying the current setting

\$>JAIR, MAN, NORM

\$>JAIR, MAN, HIGH

\$>JAIR, MAN, LOW

\$>JAIR, AUTO, NORM

Example To set the AIR mode to LOW issue the following command:

\$JAIR, LOW<CR><LF>

The response is then:

\$>JAIR, MAN, LOW<CR><LF>

Additional Information Defaults to normal (NORM) which is recommended for most applications. The AUTO option enables the receiver to decide when to turn JAIR to HIGH.

CAUTION: Setting AIR mode to HIGH is not recommended for Crescent Vector operation.

On the HIGH setting, the receiver tolerates larger and sudden drops in the SNR value before it discards the data as being invalid. This additional tolerance is beneficial in applications such as crop dusting where an aircraft is banking rapidly. As the aircraft banks, the antenna position shifts from upright and having a clear view of the sky to being tipped slightly, with a possibly obscured view of the sky, and then back to upright.

This sudden tipping of the antenna causes the SNR value to drop.

If the tolerance is not set as HIGH, the receiver views the data recorded while banking as invalid and discards it. As a result the GPS position will not be accurate.

The status of this command is also output in the [JSHOW](#) message.

JALT Command

Command Type [General Operation and Configuration](#)

Description Turn altitude aiding for the receiver on or off

When set to something other than NEVER, altitude aiding uses a fixed altitude instead of using one satellite's observations to calculate the altitude. The advantage of this feature, when operating in an application where a fixed altitude is acceptable, is that the extra satellite's observations can be used to the betterment of the latitude, longitude, and time offset calculations, resulting in improved accuracy and integrity. Marine markets, for example, may be well suited for use of this feature.

Command Format \$JALT,c[,h[,GEOID]]<CR><LF>

where 'c' (feature status variable) and 'h' (threshold variable) may be one of the following:

c Value	Description	Corresponding h Value
NEVER	Default mode of operation where altitude aiding is not used	N/A
SOMETIMES	Sets the receiver to use altitude aiding (depending upon the PDOP threshold) specified by 'h'	See following Note
SATS	Sets the receiver to use altitude aiding depending upon the number of visible satellites. If there are fewer visible satellites than specified by 'h', altitude aiding will be used.	
ALWAYS	Sets the receiver to use altitude aiding regardless of a variable. In this case, the ellipsoidal altitude 'h' that the receiver should use may be specified.	

To get an 'h' value to use with SOMETIMES and ALWAYS, using DGPS positions, average the height over a period of time (the longer the time period, the more accurate this height value).

\$JALT,ALWAYS,h<CR><LF>

In this command 'h' is the ellipsoid height.

If you use the height reported from the [GPGGA](#) message, because this is actually geoidal and not ellipsoidal height, use:

\$JALT,ALWAYS,h,GEOID<CR><LF>

Receiver Response \$>

Example To turn altitude aiding on to SOMETIMES with an ellipsoidal height of 404.2 m issue the following command:

```
$JALT,SOMETIMES,404.2<CR><LF>
```

To turn altitude aiding on to ALWAYS using the height of 401.6 m as reported in the GPGGA message (geoidal height) issue the following command:

```
$JALT,ALWAYS,401.6,GEOID<CR><LF>
```

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command. For example, if you issue the following command:

```
$JALT,SOMETIMES,404.2<CR><LF>
```

...then issuing the JSHOW command displays the following as part of its output:

```
$>JSHOW,ALT,SOMETIMES,404.2
```

JAPP Command

Command Type [General Operation and Configuration](#)

Description Specify which of the installed applications should be utilized or query the receiver for the currently installed applications

Note: Hemisphere GPS Crescent and Eclipse GPS receivers are able to hold up to two different application firmware programs simultaneously.

Command Format Specify receiver application firmware (when two applications are present)

\$JAPP , OTHER<CR><LF>
or
\$JAPP , APP<CR><LF>

where 'APP' may be one of the following by name (depending on the underlined receiver type):

Crescent

- WAAS – Changes to the SBAS application. For the sake of the application names, the SBAS application is referred to as WAAS by the receiver's internal firmware
- AUTODIFF – Changes to the e-Dif application. Referred to as "AUTODIFF" in the receiver's internal firmware
- LOCRTK – Changes to the local differential rover application
- RTKBAS – Changes to the local differential base application
- LBAND – Changes to OmniSTAR VBS

Eclipse

- WAASRTKB – Changes to the SBAS/RTK Base application
- OMNIHP – Changes to OmniSTAR XP/HP application
- RTK – Changes to the RTK Rover application

Eclipse II

- SBASRTKB – Changes to the SBAS/OmniSTAR/RTK Base application
- AUTODIFF – Changes to the e-Dif application, referred to as "AUTODIFF" in the firmware
- RTK – Changes to the RTK Rover application

miniEclipse

- WAASRTKB – Changes to the SBAS/RTK Base application
- AUTODIFF – Changes to the e-Dif application, referred to as "AUTODIFF" in the firmware
- RTK – Changes to the RTK Rover application

Query receiver application firmware

```
$JAPP<CR><LF>
```

Receiver Response

For example, if WAAS (SBAS) and AUTODIFF (e-Dif) are the two installed applications (WAAS in slot1 and AUTODIFF in slot2) and WAAS is the current application, if you issue the `$JAPP,OTHER<CR><LF>` command on a receiver, the response to `$JAPP<CR><LF>` will be `$>JAPP,AUTODIFF,WAAS,2,1,` indicating that application slot 2 (e-Dif) is currently being used.

Other derivatives of the `$JAPP` command are the `$JAPP,1<CR><LF>` and `$JAPP,2<CR><LF>` commands. You can use these to set the receiver to use the first or second application.

Hemisphere GPS recommends that you follow up the sending of these commands with a `$JAPP` query to see which application is 1 or 2. It is best to use these two commands when upgrading the firmware inside the receiver, because the firmware upgrading utility uses the application number to designate which application to overwrite.

Response to querying the current setting

```
$>JAPP,CURRENT,OTHER,[1 OR 2],[2 OR 1]
```

where:

- 'CURRENT' indicates the current application in use
- 'OTHER' indicates the secondary application that is not currently in use
- 1 and 2 indicate which application slot is currently being used

For example, if the response to `$JAPP<CR><LF>` is `$>JAPP,WAAS,AUTODIFF,1,2,` it indicates that:

- WAAS (SBAS) is in application slot 1
- e-Dif is in application slot 2
- WAAS in application slot 1 is currently being used

Example

If WAASRTKB is the current application and OMNIHP is the other (second application) the response is then:

```
$>JAPP,WAASRTKB,OMNIHP,1,2<CR><LF>
```

Additional Information

JASC Command Overview

The JASC command is used to request ASCII messages.

Command	Description
<u>JASC.CMR</u>	Set the proprietary CMR messages to on or off to provide corrections to the rover
<u>JASC,D1 (RD1)</u>	Set the RD1 diagnostic information message from the receiver to on or off
<u>JASC.DFX</u>	Set the proprietary DFX messages to on or off to provide corrections to the rover
<u>JASC.GL</u>	Enable the GLONASS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.
<u>JASC.GN</u>	Enable the GNSS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.
<u>JASC.GP</u>	Enable the GPS data messages at a particular update rate to be turned on or off
<u>JASC.INTLT</u>	Configure the receiver to output pitch and roll data
<u>JASC.PASHR</u>	Configure the receiver to output time, heading, roll, and pitch data in one message
<u>JASC.PSAT.RTKSTAT</u>	Configure the receiver to output the most relevant parameters affecting RTK
<u>JASC.PTSS1</u>	Configure the receiver to output heading, pitch, roll, and heave in the commonly used TSS1 message format
<u>JASC.ROX</u>	Set the proprietary ROX messages to on or off to provide corrections to the rover
<u>JASC.RTCM</u>	Configure the receiver to output RTCM version 2 DGPS corrections from SBAS or beacon through either receiver serial port
<u>JASC.RTCM3</u>	Set the RTCM version 3 messages to on or off to provide corrections to the rover
<u>JASC.VIRTUAL</u>	Configure the receiver to have RTCM data input on one port and output through the other (when using an external correction source)

JASC,CMR Command

Command Type [Local Differential and RTK](#)

Description Set the proprietary CMR messages to on or off to provide corrections to the rover

This command only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode. RTK is relative to the reference position (base only).

Command Format \$JASC,CMR,R[,OTHER] <CR><LF>

where:

- 'R' = correction status variable (0 = turn corrections Off, 1 = turn corrections On)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$>

Example To turn on CMR messages on the OTHER port issue the following command:

\$JASC,CMR,1,OTHER<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command.

To change the broadcast station ID, use [JRTK.28](#).

JASC,D1 Command

Command Type [General Operation and Configuration](#), [SBAS](#)

Description Set the RD1 diagnostic information message from the receiver to on or off
There is currently only an (R)D1 message.

Command Format \$JASC,D1,R[,OTHER]<CR><LF>

where:

- 'R' = message rate (0 = Off, 1 = On at 1Hz)
- ',OTHER' = optional field, enacts a change in the [RD1 message](#) on the current port when you send the command without it (and without the brackets) and enacts a change in the RD1 message on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$>

Example To output the RD1 message once per second from THIS port issue the following command:

```
$JASC,D1,1<CR><LF>
```

...and the output will look similar to the following:

```
$RD1,410213,1052,1551.489,1,0,39,-611.5,0,1F,1F,0,999999
$RD1,410214,1052,1551.489,1,0,40,-615.1,0,1F,1F,0,999999
$RD1,410215,1052,1551.489,1,0,40,-607.1,0,1F,1F,0,999999
```

See [RD1 message](#) for a description of each field in the response.

Additional Information Although you request D1 through this command the responding message is RD1.

To query the receiver for the current setting, issue the [JSHOW](#) command. For example, if you issue the following command:

```
$JASC,D1,1<CR><LF>
```

...then issuing the JSHOW command displays the following as part of its output:

```
$>JSHOW,ASC,D1,1
```

JASC,DFX Command

Command Type [Local Differential and RTK](#)

Description Set the proprietary DFX messages to on or off to provide corrections to the rover

This command only applies to a Crescent base receiver when using L-Dif or RTK mode. Differential is relative to the reference position (base only). See the [JASC,ROX](#) command for the equivalent message for the Eclipse series of products.

Command Format \$JASC,DFX,R[,OTHER]<CR><LF>

where:

- 'R' = correction status variable (0 = turn corrections Off, 1 = turn corrections On)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$>

Example To turn on DFX messages on THIS port issue the following command:

\$JASC,DFX,1<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command.

To change the broadcast station ID, use [JRTK,28](#).

JASC,GL Command

Command Type [GLONASS](#)

Description Enable the GLONASS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.

Command Format \$JASC,MSG,R[,OTHER]<CR><LF>

where

- 'MSG' = name of the data message
- 'R' = message rate (see table below)
- ',OTHER' = optional field, enacts a change on the current port (THIS port) when you send the command without it (and without the brackets) and enacts a change on the other port (OTHER port) when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Send a command with a zero value for the 'R' field to turn off a message.

MSG	R (rate in Hz)	Description
GLMLA	1 or 0	GLONASS almanac data
GLGGA	20, 10, 2, 1, 0 or .2	GPS fix data
GLGLL	20, 10, 2, 1, 0 or .2	Geographic position - latitude/longitude
GLGNS	20, 10, 2, 1, 0 or .2	GNSS fix data
GLGSA	1 or 0	GLONASS DOP and active satellites
GLGSV	1 or 0	GLONASS satellite in view

Receiver Response \$>

Example To output the GLGNS message through the OTHER port at a rate of 20 Hz, issue the following command:

\$JASC,GLGNS,20,OTHER<CR><LF>

Additional Information The status of this command is also output in the [JSHOW](#) message.
[What does <CR><LF> mean?](#)

JASC,GN Command

Command Type [GPS](#), [Crescent Vector](#)

Description Enable the GNSS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.

Command Format \$JASC ,MSG ,R[, OTHER] <CR><LF>

where

- 'MSG' = name of the data message
- 'R' = message rate (see table below)
- ',OTHER' = optional field, enacts a change on the current port (THIS port) when you send the command without it (and without the brackets) and enacts a change on the other port (OTHER port) when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Send a command with a zero value for the 'R' field to turn off a message.

MSG	R (rate in Hz)	Description
GPGGA	20, 10, 2, 1, 0 or .2	GPS fix data
GPGLL	20, 10, 2, 1, 0 or .2	Geographic position - latitude/longitude
GPGNS	20, 10, 2, 1, 0 or .2	GNSS fix data
GPGSA	1 or 0	GNSS DOP and active satellites

Receiver Response \$>

Example To output the GNGNS message through the OTHER port at a rate of 20 Hz, issue the following command:
\$JASC ,GNGNS , 20 , OTHER<CR><LF>

Additional Information The status of this command is also output in the [JSHOW](#) message.
[What does <CR><LF> mean?](#)

JASC,GP Command

Command Type [GPS](#), [Crescent Vector](#)

Description Enable the GPS data messages at a particular update rate to be turned on or off. When turning messages on, various update rates are available depending on the requirements.

Command Format \$JASC,MSG,R[,OTHER] <CR><LF>

where

- 'MSG' = name of the data message
- 'R' = message rate (see table below)
- ',OTHER' = optional field, enacts a change on the current port (THIS port) when you send the command without it (and without the brackets) and enacts a change on the other port (OTHER port) when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Send a command with a zero value for the 'R' field to turn off a message.

MSG	R (rate in Hz)	Description
GPALM	1 or 0	GPS almanac data
GPD TM	1 or 0	Datum reference
GPGBS	1 or 0	Satellite fault detection used for RAIM
GPGGA	20, 10, 2, 1, 0 or .2	Detailed GPS position information
GPGLL	20, 10, 2, 1, 0 or .2	Latitude and longitude data
GPGNS	20, 10, 2, 1, 0 or .2	Fixes data for single or combined satellite navigation systems
GPGRS	20, 10, 2, 1, 0 or .2	GNSS range residuals
GPGSA	1 or 0	GPS DOP and active satellite information
GPGST	1 or 0	GNSS pseudorange error statistics
GPGSV	1 or 0	GNSS satellite in view
GPHDG or HEHDG	20, 10, 2, 1, 0 or .2	Magnetic deviation and variation for calculating magnetic or true heading
GPHDM or HEHDM	20, 10, 2, 1, 0 or .2	Magnetic heading of the vessel derived from the true heading calculated
GPHDT or HEHDT	20, 10, 2, 1, 0 or .2	True heading of the vessel
GPHEV	20, 10, 2, 1, 0 or .2	Heave value in meters

GPHPR	20, 10, 2, 1, 0 or .2	Proprietary NMEA message that provides the heading, pitch, roll, and time in a single message
GPRMC	10, 2, 1, 0 or .2	Recommended minimum specific GNSS data
GPROT or HEROT	20, 10, 2, 1, 0 or .2	Vessel's rate of turn (ROT) information
GPRRE	1 or 0	Range residual message
GPVTG	20, 10, 2, 1, 0 or .2	Course over ground and ground speed
GPZDA	20, 10, 2, 1, 0 or .2	UTC time and date information
INTLT	1 or 0	Proprietary NMEA message that provides the tilt measurements from the internal inclinometers (in degrees)

Receiver Response \$>

Example To output the GPGGA message through the OTHER port at a rate of 20 Hz, issue the following command:
\$JASC , GPGGA , 20 , OTHER<CR><LF>

Additional Information The status of this command is also output in the [JSHOW](#) message.
[What does <CR><LF> mean?](#)

JASC,INTLT Command

Command Type [Crescent Vector](#)

Description Configure the receiver to output pitch and roll data (pitch and roll are factory calibrated over temperature to be accurate to $\pm 3^{\circ}\text{C}$)

Command Format `$JASC,INTLT,R[,OTHER]<CR><LF>`
where

- 'R' = message rate (0 = Off, 1 = On at 1Hz)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response `$PSAT,INTLT,pitch,roll*CC<CR><LF>`
where pitch and roll are in degrees

Example

Additional Information [PSAT,INTLT](#) message

JASC,PASHR Command

Command Type [Crescent Vector](#)

Description Configure the receiver to output time, heading, roll, and pitch data in one message

Command Format \$JASC,PASHR,R[,OTHER]<CR><LF>

where

- 'R' = message rate (0 = Off, 1 = On at 1Hz)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$PASHR,hhmmss.ss,HHH.HH,T,RRR.RR,PPP.PP,heave,rr.rrr,pp.ppp,hh.hhh,QF*CC<CR><LF>

where:

Message Component	Description
hhmmss.ss	UTC time
HHH.HH	Heading value in decimal degrees
T	True heading (T displayed if heading is relative to true north)
RRR.RR	Roll in decimal degrees (- sign will be displayed when applicable)
PPP.PP	Pitch in decimal degrees (- sign will be displayed when applicable)
heave	Heave, in meters
rr.rrr	Roll standard deviation in decimal degrees
pp.ppp	Pitch standard deviation in decimal degrees
hh.hhh	Heading standard deviation in decimal degrees
QF	Quality Flag <ul style="list-style-type: none"> • 0 = No position • 1 = All non-RTK fixed integer positions • 2 = RTK fixed integer position
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Example To turn on the PASHR message on THIS port issue the following command:

```
$JASC , PASHR , 1 <CR> <LF>
```

...and the message output appears similar to the following:

```
$PASHR,162930.00,,T,2.48,3.92,-0.64,0.514,0.514,0.000,1*05  
$PASHR,162931.00,,T,2.38,3.93,-0.70,0.508,0.508,0.000,1*07  
$PASHR,162932.00,,T,2.67,4.00,-0.66,0.503,0.503,0.000,1*04
```

**Additional
Information** [PASHR](#) message

JASC,PSAT,RTKSTAT Command

Command Type [Local Differential and RTK](#)

Description Configure the receiver to output the most relevant parameters affecting RTK

Command Format \$JASC,PSAT,RTKSTAT,R[,OTHER]<CR><LF>

where:

- 'R' = message rate (0 = Off, 1 = On at 1Hz)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

You can also perform a one-time query of the message information by issuing the [JQUERY,RTKSTAT](#) command.

Receiver Response \$>

Example To turn on this message on the THIS port issue the following command:

\$JASC,PSAT,RTKSTAT,1<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command. See also [PSAT,RTKSTAT](#) message.

JASC,PTSS1 Command

Command Type [Crescent Vector](#)

Description Configure the receiver to output heading, pitch, roll, and heave in the commonly used TSS1 message format

Command Format \$JASC,PTSS1,R[,OTHER]<CR><LF>

where

- 'R' = message rate (in Hz) of 0 (off), 0.25, 0.5, 1, 2, 4, 5, 10, or 20 (if subscribed)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response :XXXXAASMHQHQMRRRRSMPPPP*CC<CR><LF>

where:

Messa ge Compo nent	Description						
XX	Horizontal acceleration						
AAAA	Vertical acceleration						
HHHH	Heave						
S	S = space character						
M	Space if positive; minus if negative						
Q	Status flag <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>h</td><td>Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.</td></tr> <tr> <td>F</td><td>Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.</td></tr> </table>	Value	Description	h	Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.	F	Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.
Value	Description						
h	Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.						
F	Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.						
M	Space if positive; minus if negative						
RRRR	Roll						

S	S = space character
M	Space if positive; minus if negative
PPPP	Pitch
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

**Additional
Information** [TSS1](#) message

JASC,ROX Command

Command Type [Local Differential and RTK](#)

Description Set the proprietary ROX messages to on or off to provide corrections to the rover

This command only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode. RTK is relative to the reference position (base only).

Command Format \$JASC,ROX,R[,OTHER]<CR><LF>

where:

- 'R' = correction status variable (0 = turn corrections Off, 1 = turn corrections On)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$>

Example To turn on ROX messages on the OTHER port issue the following command:

\$JASC,ROX,1,OTHER<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command.

To change the broadcast station ID, use [JRTK.28](#).

JASC,RTCM Command

Command Type [SBAS](#)

Description Configure the receiver to output RTCM version 2 DGPS corrections from SBAS or beacon through either receiver serial port. The correction data output is RTCM SC-104, even though SBAS uses a different over-the-air protocol (RTCA).

Command Format \$JASC,RTCM,R[,OTHER]<CR><LF>

where:

- 'R' = message status variable (0 = Off, 1 = On)
 - ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.
-

Receiver Response \$>

Example To output RTCM corrections from SBAS or beacon on THIS port (current port) issue the following command:

\$JASC,RTCM,1<CR><LF>

Additional Information To verify the current setting is on, issue the [JSHOW](#) command. You will see output similar to the following:

\$>JSHOW,ASC,RTCM,1.0

If the current setting is off, the JSHOW command will not show any information for this setting.

JASC,RTCM3 Command

Command Type [Local Differential and RTK](#)

Description Set the RTCM version 3 messages to on or off to provide corrections to the rover

This command only applies to an Eclipse base station receiver when using GPS dual frequency RTK mode. RTK is relative to the reference position (base only).

Command Format \$JASC,RTCM3,R[,OTHER]<CR><LF>

where:

- 'R' = correction status variable (0 = turn corrections Off, 1 = turn corrections On)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Receiver Response \$>

Example To turn on RTCM3 messages on the OTHER port issue the following command:

\$JASC,RTCM3,1,OTHER<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command.

To change the broadcast station ID, use [JRTK.28](#).

JASC,VIRTUAL Command

Command Type [General Operation and Configuration](#)

Description Configure the receiver to have RTCM data input on one port and output through the other (when using an external correction source)

For example, if RTCM is input on Port B, the data will be output through Port A having corrected the receiver position. The receiver acts as a pass-through for the RTCM data. Either port may be configured to accept RTCM data input; this command enables the opposite port to output the RTCM data.

Command Format \$JASC,VIRTUAL,R[,OTHER]<CR><LF>

where:

- 'R' = message status variable (0 = Off, 1 = On)
 - ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.
-

Receiver Response \$>

Example To configure THIS port to output RTCM messages that are being input through the OTHER port issue the following command:

\$JASC,VIRTUAL,1

Additional Information

JATT Command Overview

The JATT command is used to define or query attitude settings for Vector products.

Command	Description
JATT,COGTAU	Set the course over ground (COG) time constant (0.0 to 3600.0 seconds) or query the current setting
JATT,CSEP	Query to retrieve the current separation between GPS antennas
JATT,EXACT	Enable/disable internal filter reliance on the entered antenna separation or query the current setting
JATT,FLIPBRD	Allow upside down installation
JATT,GYROAID	Turn on gyro aiding or query the current feature status
JATT,HBIAS	Set the heading bias or query the current setting
JATT,HELP	Show the available commands for GPS heading operation and status
JATT,HIGHMP	Set/query the high multipath setting for use in poor GPS environments
JATT,HRTAU	Set the rate of turn time constant or query the current setting
JATT,HTAU	Set the heading time constant or query the current setting
JATT,LEVEL	Turn on level operation or query the current feature status
JATT,MSEP	Set (manually) the GPS antenna separation or query the current setting
JATT,NEGILT	Turn on the negative tilt feature or query the current setting
JATT,NMEAHE	Instruct the Crescent Vector on how to preface the HDT and HDR messages
JATT,PBIAS	Set the pitch bias or query the current setting
JATT,PTAU	Set the pitch time constant or query the current setting
JATT,ROLL	Configure the Crescent Vector for roll or pitch output
JATT,SEARCH	Force a new RTK heading search
JATT,SPDTAU	Set the speed time constant (0.0 to 3600.0 seconds) or query the current setting
JATT,SUMMARY	Show the current configuration of the Crescent Vector
JATT,TILTAID	Turn tilt aiding on/off or query the Crescent Vector for the current status of this feature
JATT,TILTCAL	Calibrate the internal tilt sensor of the Crescent Vector

JATT,CSEP Command

Command Type [Crescent Vector](#)

Description Query the Crescent Vector for the current calculated separation between antennas, as solved for by the attitude algorithms

Command Format \$JATT , CSEP<CR><LF>

Receiver Response \$>JATT , X , CSEP
where 'X' is the antenna separation in meters

Additional Information

JATT,COGTAU Command

Note: The [JTAU,COG](#) command provides identical functionality but works with Crescent and Eclipse products in addition to Crescent Vector products.

Command Type [Crescent Vector](#)

Description Set the course over ground (COG) time constant (0.0 to 3600.0 seconds) or query the current setting

This command allows you to adjust the level of responsiveness of the COG measurement provided in the [GPVTG](#) message. The default value is 0.0 seconds of smoothing. Increasing the COG time constant increases the level of COG smoothing.

Command Format Set the COG time constant

```
$JATT,COGTAU,cogtau<CR><LF>
```

where "cogtau" is the new COG time constant that falls within the range of 0.0 to 200.0 seconds

The setting of this value depends upon the expected dynamics of the Crescent. If the Crescent will be in a highly dynamic environment, this value should be set lower because the filtering window would be shorter, resulting in a more responsive measurement. However, if the receiver will be in a largely static environment, this value can be increased to reduce measurement noise.

Query the current setting

```
$JATT,COGTAU<CR><LF>
```

Receiver Response \$>

Additional Information You can use the following formula to determine the COG time constant:

$\text{cogtau (in seconds)} = 10 / \text{maximum rate of change of course (in } ^\circ/\text{s)}$

If you are unsure about the best value for this setting, it is best to be conservative and leave it at the default setting of 0.0 seconds.

JATT,EXACT Command

Command Type [Crescent Vector](#)

Description	Enable/disable internal filter reliance on the entered antenna separation or query the current setting
--------------------	--

Command Format	Enable/disable internal filter reliance
-----------------------	---

To enable internal filter reliance:

\$JATT , EXACT , YES<CR><LF>

To disable internal filter reliance:

\$JATT , EXACT , NO<CR><LF>

Query the current setting

\$JATT , EXACT<CR><LF>

Receiver Response	\$>
--------------------------	-----

Additional Information

JATT,FLIPBRD Command

Command Type [Crescent Vector](#)

Description Turn the flip feature on/off or query the current feature status

Allow the Crescent Vector OEM board to be installed upside down. You should use this command only with the Vector Sensor and the Crescent Vector OEM board because flipping the OEM board does not affect the antenna array that needs to remain facing upwards. When using this command, the board needs to be flipped about roll so the front still faces the front of the vessel.

Command Format Turn the flip feature on/off

To turn the flip feature on:

```
$JATT,FLIPBRD,YES<CR><LF>
```

To turn the flip feature off (return to default mode - right side up):

```
$JATT,FLIPBRD,NO<CR><LF>
```

Query current the current setting

```
$JATT,FLIPBRD<CR><LF>
```

Receiver Response \$>

Additional Information

JATT,GYROAID Command

Command Type [Crescent Vector](#)

Description Turn gyro aiding on or off or query the current setting

The Crescent Vector's internal gyro—enabled by default when shipped—offers two benefits.

- It shortens reacquisition times when a GPS heading is lost because of obstruction of satellite signals. It does this by reducing the search volume required for solution of the RTK.
- It provides an accurate substitute heading for a short period (depending on the roll and pitch of the vessel) ideally seeing the system through to reacquisition.

For these two benefits, Hemisphere GPS highly recommend leaving gyro aiding on.

Exceeding rates of 90°/sec is not recommended because the gyro cannot measure rates beyond this point. This is a new recommendation since Hemisphere GPS now uses gyro measurements to obtain a heading rate measurement.

Command Format Turn gyro aiding on/off

To turn gyro aiding on:

\$JATT,GYROAID,YES<CR><LF>

To turn gyro aiding off:

\$JATT,GYROAID,NO<CR><LF>

Query the current setting

\$JATT,GYROAID<CR><LF>

Receiver Response \$>

Additional Information Every time you power up the Crescent Vector the gyro goes through a warmup procedure and calibrates itself. You cannot save the resulting calibration, so the self-calibration takes place every time the Crescent Vector is power cycled.

This self-calibration procedure takes several minutes and is the equivalent of the following manual calibration procedure.

With the Crescent Vector unit installed:

1. Apply power and wait several minutes until it has acquired a GPS signal and is computing heading.
 2. Ensure gyroaiding is on by issuing the following command:
\$JATT,GYROAID<CR><LF>
 3. Slowly spin the unit for one minute at no more than 15°/sec.
 4. Keep the unit stationary for four minutes. Both the manual and the self-calibration procedures calibrate the Crescent Vector's gyro to the same effect.
-

JATT,HBIAS Command

Command Type [Crescent Vector](#)

Description Set the heading output from the Crescent Vector to calibrate the true heading of the antenna array to reflect the true heading of the vessel or query the current setting

Command Format Set the heading output

\$JATT , HBIAS , X<CR><LF>

where 'X' is a bias that will be added to the Crescent Vector's heading in degrees. The acceptable range for the heading bias is -180.0° to 180.0°. The default value of this feature is 0.0°.

Query the current setting (current compensation angle)

\$JATT , HBIAS<CR><LF>

Receiver Response \$>

Additional Information

JATT,HELP Command

Command Type [Crescent Vector](#)

Description Show the available commands for GPS heading operation and status

Command Format \$JATT,HELP<CR><LF>

Receiver Response \$>JATT,HELP,CSEP,MSEP,EXACT,LEVEL,HTAU,HRTAU,HBIASPBias,NEGTILT,ROLL,TILTAID,TILTCAL,MAGCID,MAGCAL,MAGCLR,GYROCID,COGTAU,SPDTAU,SEARCH,SUMMARY

Additional Information

JATT,HIGHMP Command

Command Type [Crescent Vector](#)

Description Enable/disable the high multipath setting for use in poor GPS environments or query the current setting

Enabling HIGHMP mode may result in longer heading acquisition times in high multipath environments. In HIGHMP mode, the Vector will not output heading until it has good confidence in the result. In very poor environments, this may take a few minutes or more; in normal environments, there is only a slight increase in heading acquisition time.

Command Format Set the high multipath setting

To enable the high multipath setting:

```
$JATT,HIGHMP,YES<CR><LF>
```

To disable the high multipath setting:

```
$JATT,HIGHMP,NO<CR><LF>
```

Query the current setting

```
$JATT,HIGHMP<CR><LF>
```

Receiver Response \$>

Additional Information

JATT,HRTAU Command

Command Type [Crescent Vector](#)

Description Set the heading rate time constant to adjust the level of responsiveness of the rate of heading change measurement provided in the [GPROT](#) message or query the current setting

The default value of this constant is 2.0 seconds of smoothing. Increasing the time constant increases the level of heading smoothing.

Command Format Set the heading rate time constant

\$JATT,HRTAU,hrtau<CR><LF>

where 'hrtau' is the new time constant that falls within the range of 0.0 to 3600.0 seconds

The setting of this value depends upon the expected dynamics of the vessel. For example, if the vessel is very large and cannot turn quickly, increasing this time is reasonable. The resulting heading would have reduced 'noise', resulting in consistent values with time. However, artificially increasing this value such that it does not agree with a more dynamic vessel could create a lag in the rate of heading change measurement with higher rates of turn.

Query the current setting

\$JATT,HRTAU<CR><LF>

Receiver Response \$>

Additional Information You can use the following formula to determine the level of smoothing:

$\text{hrtau (in seconds)} = 10 / \text{maximum rate of the rate of turn (in } ^\circ/\text{s}^2)$

Note: If you are unsure about the best value for the setting, leave it at the default setting of 2.0 seconds.

JATT,HTAU Command

Command Type [Crescent Vector](#)

Description Set the heading time constant to adjust the level of responsiveness of the true heading measurement provided in the [GPHDT](#) message or query the current setting

The default value of this constant is 2.0 seconds of smoothing when the gyro is enabled.

Although, the gyro is enabled by default, you can turn it off. When you turn the gyro off, the default value of the heading time constant is 0.5 seconds of smoothing. Increasing the heading time constant increases the level of heading smoothing.

Command Format Set the heading time constant

```
$JATT,HTAU,htau<CR><LF>
```

where 'htau' is the new time constant that falls within the range of 0.0 to 3600.0 seconds

The setting of this value depends upon the expected dynamics of the vessel. If the vessel is very large and cannot turn quickly, increasing this time is reasonable. The resulting heading would have reduced 'noise' resulting in consistent values with time. However, artificially increasing this value such that it does not agree with a more dynamic vessel could create a lag in the heading measurement with higher rates of turn.

Query the current setting

```
$JATT,HTAU<CR><LF>
```

Receiver Response \$>

Additional Information You can use the following formula to determine level of heading smoothing required when the gyro is in use:

Gyro on
 $\text{htau (in seconds)} = 40 / \text{maximum rate of turn (in } ^\circ/\text{s)}$

Gyro off
 $\text{htau (in seconds)} = 10 / \text{maximum rate of turn (in } ^\circ/\text{s)}$

If you are unsure about the best value for the setting, leave it at the default setting of 2.0 seconds when the gyro is on and at 0.5 seconds when the gyro is off.

JATT,LEVEL Command

Command Type [Crescent Vector](#)

Description Turn level operation on or off or query the current setting

Invoke the level operation mode of the Crescent Vector. If the application will not involve the system tilting more than $\pm 10^\circ$, you may use this mode of operation. The benefit of using level operation is increased robustness and faster acquisition times of the RTK heading solution.

This feature is turned off by default.

Command Format Turn level operation on/off

To turn level operation on:

`$JATT,LEVEL,YES<CR><LF>`

To turn level operation off:

`$JATT,LEVEL,NO<CR><LF>`

Query the current setting

`$JATT,LEVEL<CR><LF>`

Receiver Response `$>`

Additional Information

JATT,MSEP Command

Command Type [Crescent Vector](#)

Description Manually enter a custom separation between antennas (must be accurate to within 1 to 2 cm) or query the current setting

Command Format Set the antenna separation

Using the new center-to-center measurement, issue the following command:

```
$JATT,MSEP,sep<CR><LF>
```

where "sep" is the measured antenna separation entered in meters

Query the current setting

```
$JATT,MSEP<CR><LF>
```

Receiver Response \$>

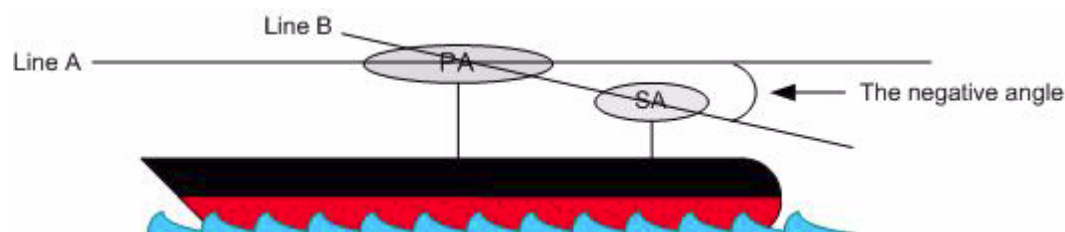
Additional Information

JATT,NEGTLT Command

Command Type [Crescent Vector](#)

Description Turn the negative tilt feature on or off or query the current setting

When the secondary GPS antenna (SA) is below the primary GPS antenna (PA), there is an angle formed between a horizontal line through the center of the primary antenna (Line A in the diagram below) and an intersecting line through the center of the primary and secondary antennas (Line B). This angle is considered to be negative.



Depending on the convention for positive and negative pitch/roll, you want to change the sign (either positive or negative) of the pitch/roll.

Command Format Turn negative tilt feature on/off

To change the sign of the pitch/roll measurement:

```
$JATT,NEGTLT,YES<CR><LF>
```

To return the sign of the pitch/roll measurement to its original value:

```
$JATT,NEGTLT,NO<CR><LF>
```

Query the current setting

```
$JATT,NEGTLT<CR><LF>
```

Receiver Response \$>

Additional Information

JATT,NMEAHE Command

Command Type [Crescent Vector](#)

Description Instruct the Crescent Vector to preface the following messages with GP or HE.

- [HDG](#)
- [HDM](#)
- [HDT](#)
- [ROT](#)

Command Format \$JATT,NMEAHE,X<CR><LF>

where 'X' is either 1 for HE or 0 for GP

To preface specific messages with GP

\$JATT,NMEAHE,0<CR><LF>

To preface specific messages with HE

\$JATT,NMEAHE,1<CR><LF>

Receiver Response \$>JATT,NMEAHE,OK

Additional Information The HDM message is for a magnetic compass. The message will be **HC**HDM when requesting with \$JATT,NMEAHE,1 specified.

JATT,PBIAS Command

Command Type [Crescent Vector](#)

Description Set the pitch/roll output from the Crescent Vector to calibrate the measurement if the antenna array is not installed in a horizontal plane or query the current setting

Command Format Set the pitch/roll output

\$JATT , PBIAS , X<CR><LF>

where "X" is a bias that will be added to the Crescent Vector's pitch/roll measure, in degrees

The acceptable range for the pitch bias is -15.0° to 15.0°. The default value is 0.0°.

Query the current setting

\$JATT , PBIAS<CR><LF>

Receiver Response \$>

Additional Information **Note:** The pitch/roll bias is added after the negation of the pitch/roll measurement (if invoked with the [JATT,NEGILT](#) command).

JATT,PTAU Command

Command Type [Crescent Vector](#)

Description Set the level of responsiveness of the pitch measurement provided in the [PSAT,HPR](#) message or query the current setting

The default value of the pitch time constant is 0.5 seconds of smoothing. Increasing the pitch time constant increases the level of pitch smoothing.

Command Format Set the pitch time constant

```
$JATT,PTAU,ptau<CR><LF>
```

where 'ptau' is the new time constant that falls within the range of 0.0 to 3600.0 seconds

The setting of this value depends upon the expected dynamics of the vessel. For instance, if the vessel is very large and cannot pitch quickly, increasing this time is reasonable. The resulting pitch would have reduced 'noise', resulting in consistent values with time. However, artificially increasing this value such that it does not agree with a more dynamic vessel could create a lag in the pitch measurement.

Query the current setting

```
$JATT,PTAU<CR><LF>
```

Note: If you are unsure about the best value for the setting, leave it at the default setting of 0.5 seconds

Receiver Response \$>

Additional Information You can use the following formula to determine the level of pitch smoothing required:

$$\text{ptau (in seconds)} = 10 / \text{maximum rate of pitch (in } ^\circ/\text{s)}$$

JATT,ROLL Command

Command Type [Crescent Vector](#)

Description Configure the Crescent Vector for roll or pitch GPS antenna orientation

Command Format Configure the Crescent Vector for pitch or roll GPS antenna orientation

To configure the Crescent Vector for roll GPS antenna orientation (the Antenna Array must be installed perpendicular to the vessel's axis):

```
$JATT,ROLL,YES<CR><LF>
```

To configure the Crescent Vector for pitch GPS antenna orientation (default):

```
$JATT,ROLL,NO<CR><LF>
```

Query the current setting

```
$JATT,ROLL<CR><LF>
```

Receiver Response \$>

Additional Information

JATT,SEARCH Command

Command Type [Crescent Vector](#)

Description Force the Crescent Vector to reject the current GPS heading solution and begin a new search

Command Format \$JATT,SEARCH<CR><LF>

Receiver Response \$>

Additional Information The SEARCH function will not work if you have enabled the gyroaid feature (using the [GYROAID](#) command). In this case you must cycle power to the receiver to have a new GPS solution computed.

JATT,SPDTAU Command

Note: The [JTAU,SPEED](#) command provides identical functionality but works with Crescent and Eclipse products in addition to Crescent Vector products.

Command Type [Crescent Vector](#)

Description Set the speed time constant (0.0 to 3600.0 seconds) or query the current setting

This command allows you to adjust the level of responsiveness of the speed measurement provided in the [GPVTG](#) message. The default value is 0.0 seconds of smoothing. Increasing the speed time constant increases the level of speed measurement smoothing.

Command Format Set the speed time constant

\$JATT,SPDTAU,spdtau<CR><LF>

where 'spdtau' is the new time constant that falls within the range of 0.0 to 200.0 seconds

The setting of this value depends upon the expected dynamics of the receiver. If the receiver will be in a highly dynamic environment, you should set this to a lower value, since the filtering window will be shorter, resulting in a more responsive measurement. However, if the receiver will be in a largely static environment, you can increase this value to reduce measurement noise.

Query the current setting

\$JATT,SPDTAU<CR><LF>

Receiver Response \$>

Additional Information You can use the following formula to determine the COG time constant (Hemisphere GPS recommends testing how the revised value works in practice):

$$\text{spdtau (in seconds)} = 10 / \text{maximum acceleration (in m/s}^2\text{)}$$

If you are unsure of the best value for this setting, it is best to be conservative and leave it at the default setting:

- Crescent Vector receivers: default of 0.0 seconds
 - Non-Crescent Vector receivers: default of LONG (900 seconds)
-

JATT,SUMMARY Command

Command Type [Crescent Vector](#)

Description Display a summary of the current Crescent Vector settings

Command Format \$JATT , SUMMARY<CR><LF>

Receiver Response \$>JATT , SUMMARY , htau , hrtau , ptau , cogtau , spdtau , hbias , pbias , hexflag<CR><LF>

where:

Component	Description																		
htau	Current heading time constant, in seconds																		
hrtau	Current heading rate time constant, in seconds																		
ptau	Current pitch time constant, in seconds																		
cogtau	Current course over ground time constant, in seconds																		
spdtau	Current speed time constant, in seconds																		
hbias	Current heading bias, in degrees																		
pbias	Current pitch/roll bias, in degrees																		
hexflag	<div>Hex code that summarizes the heading feature status</div> <table><tr><th><u>Flag</u></th><th><u>'On'</u> <u>Value</u></th><th><u>'Off'</u> <u>Value</u></th></tr><tr><td>Gyro aiding</td><td>02</td><td>0</td></tr><tr><td>Negative tilt</td><td>01</td><td>0</td></tr><tr><td>Roll</td><td>08</td><td>0</td></tr><tr><td>Tilt aiding</td><td>02</td><td>0</td></tr><tr><td>Level</td><td>01</td><td>0</td></tr></table>	<u>Flag</u>	<u>'On'</u> <u>Value</u>	<u>'Off'</u> <u>Value</u>	Gyro aiding	02	0	Negative tilt	01	0	Roll	08	0	Tilt aiding	02	0	Level	01	0
<u>Flag</u>	<u>'On'</u> <u>Value</u>	<u>'Off'</u> <u>Value</u>																	
Gyro aiding	02	0																	
Negative tilt	01	0																	
Roll	08	0																	
Tilt aiding	02	0																	
Level	01	0																	

The 'hexflag' field is two separate hex flags:

- 'GN' - Value is determined by computing the sum of the gyro aiding and negative tilt values, depending on whether they are on or off:
 - If the feature is on, their value is included in the sum
 - If the feature is off, it has a value of zero when computing the sum
- 'RMTL' - Value is determined in much the same way but by adding the values of roll, tilt aiding, and level operation.

For example, if gyro aiding, roll, and tilt aiding features were each on, the values of 'GN' and 'RMTL' would be:

- 'GN' = hex (02 + 0) = hex (02) = 2
- 'RMTL' = hex (08 + 02) = hex (10) = A
- 'GN-RMTL' = 2A

The following tables summarize the possible feature configurations for the first 'GN' character and the second 'RMTL' character.

JATT,SUMMARY 1st GN Character Configurations		
GN Value	Gyro Value	Negative Tilt
0	Off	Off
1	Off	On
2	On	Off
3	On	On

JATT,SUMMARY 2nd RMTL Character Configurations			
RMTL Value	Roll	Tilt Aiding	Level
0	Off	Off	Off
1	Off	Off	On
2	Off	On	Off
3	Off	On	On
8	On	Off	Off
9	On	Off	On
A	On	On	Off
B	On	On	On

Example \$>JATT,SUMMARY,TAU:H=0.50,HR=2.00,COG=0.00,SPD=0.00,BIAS:H=0.00,P=0.00
 ,
 FLAG_HEX:HF-RMTL=01

Additional Information

JATT,TILTAID Command

Command Type [Crescent Vector](#)

Description Turn tilt aiding on or off or query the current setting

The Crescent Vector's internal tilt sensors (accelerometers) may be enabled by default (see your specific product manuals for further information).

The sensors act to reduce the RTK search volume, which improves heading startup and reacquisition times. This improves the reliability and accuracy of selecting the correct heading solution by eliminating other possible, erroneous solutions.

Command Format Turn tilt aiding on/off

Turn tilt aiding on:

```
$JATT,TILTAID,YES,<CR><LF>
```

Turn tilt aiding off:

```
$JATT,TILTAID,NO<CR><LF>
```

Query the current setting

```
$JATT,TILTAID<CR><LF>
```

Receiver Response Response to issuing command to turn tilt aiding on/off

```
$>
```

Response to querying the current setting

If setting is currently ON the response is:

```
$>JATT,TILTAID,ON
```

If setting is currently OFF the response is:

```
$>JATT,TILTAID,OFF
```

Additional Information Tilt aiding is required to increase the antenna separation of the Crescent Vector OEM beyond the default 0.5 m length.

JATT,TILTCAL Command

Command Type [Crescent Vector](#)

Description Calibrate the internal tilt sensors of the Crescent Vector

You can calibrate the tilt sensor of the Crescent Vector in the field but the Crescent Vector enclosure must be horizontal when you calibrate.

The calibration process takes about two seconds. The calibration is automatically saved to memory for subsequent power cycles.

Command Format \$JATT,TILTCAL<CR><LF>

Receiver Response \$>

Additional Information

JBAUD Command

Command Type [General Operation and Configuration](#)

Description Specify the baud rates of the receiver or query the current setting

Command Format Specify the baud rates

```
$JBAUD , R [ , OTHER ] [ , SAVE ] <CR> <LF>
```

where:

- 'R' = baud rate (4800, 9600, 19200, 38400, 57600, or 115200)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)
- ',SAVE' = optional field, saves the baud rate into flash memory so that if you reset power the receiver will boot at the new baud rate (it may take several seconds to save the baud rate to flash memory)

Query the current setting

```
$JBAUD [ , OTHER ] <CR> <LF>
```

where:

- ',OTHER' = optional field, queries the current port when you send the command without it (and without the brackets) and queries the other port when you send the command with it (without the brackets)

Receiver Response \$>JBAUD , R [, OTHER]

The response format is the same whether you specify the baud rates or query the current settings.

Example Issue the following command to set the baud rate to 19200 on the current port:

```
$JBAUD , 19200 <CR> <LF>
```

...the response is then:

```
$>JBAUD , 19200
```

Issue the following command to set the baud rate to 9600 on the OTHER port and save it into memory:

```
$JBAUD , 9600 , OTHER , SAVE <CR> <LF>
```

...the response is then:

```
$>JBAUD , 9600 , OTHER
```

**Additional
Information**

Note: When saving the baud rate wait until you see the SAVE COMPLETE message before powering off the receiver. See the [JSAVE](#) command for an example of this output.

The status of this command is also output when issuing the [JSHOW](#) command.

JBIN Command

Command Type [General Operation and Configuration](#)

Description Enable the output of the various binary messages—most notably the [Bin95](#) and [Bin96](#) messages—to be requested. The Bin95 and Bin96 messages contain all the information required for post processing.

Command Format \$JBIN,MSG,R<CR><LF>

where:

- 'MSG' = binary message you want to output
- 'R' = message rate as shown in the following table

Message Name	MSG	R (Hz)	Description
Bin1	1	20, 10, 2, 1, 0, or .2	GPS position message (position and velocity data)
Bin2	2	1 or 0	GPS DOPs (Dilution of Precision)
Bin62	62	1 or 0	GLONASS almanac information
Bin65	65	1 or 0	GLONASS ephemeris information
Bin66	66	20, 10, 2, 1, or 0	GLONASS L1 code and carrier phase information
Bin69	69	1 or 0	GLONASS L1 diagnostic information
Bin76	76	20, 10, 2, 1, 0, or .2	GPS L1/L2 code and carrier phase information
Bin80	80	1 or 0	SBAS data frame information
Bin89	89	1 or 0	SBAS satellite tracking information
Bin93	93	1 or 0	SBAS ephemeris information
Bin94	94	1 or 0	Ionospheric and UTC conversion parameters
Bin95	95	1 or 0	GPS ephemeris information
Bin96	96	20, 10, 2, 1, or 0	GPS L1 code and carrier phase information
Bin97	97	20, 10, 2, 1, 0, or .2	Processor statistics
Bin98	98	1 or 0	Satellite and almanac information
Bin99	99	1 or 0	GPS L1 diagnostic information

Receiver Response \$>

Example To output the Bin76 message at a rate of 10 Hz, issue the following command:

```
$JBIN,76,10<CR><LF>
```

Additional Information Higher update rates may be available with a subscription on Bin 1, 2, 96, 97 and 99.

JBOOT,OMNI Command

Command Type [OmniSTAR](#)

Description Power down the OmniSTAR portion of the Eclipse engine and then power it back up. This allows you to reboot the receiver to drop the satellite to which it is currently locked and retune to another satellite without cycling the power of the Eclipse II. It also allows you to reset the HP resolution algorithm.

Command Format \$JBOOT , OMNI<CR><LF>

Receiver Response \$>

Additional Information [JFREQ](#)

JCONN Command

Command Type [General Operation and Configuration](#)

Description Create a virtual circuit between two ports to enable communication through the receiver to the device on the opposite port

Command Format

To connect two ports virtually:

```
$JCONN,P1,P2<CR><LF>
```

where P1 and P2 are a pair of the following: A,B,C,D or PortA,PortB,PortC,PortD

Examples

```
$JCONN,A,B<CR><LF>
```

```
$JCONN,PortA,PortB<CR><LF>
```

To disconnect virtual connection:

```
$JCONN,X<CR><LF>
```

Receiver Response \$>

Additional Information

Caution: Hemisphere GPS receivers with menus, such as an R Series, use JCONN within the menu application. Any settings you make with JCONN on these products may disable the menu functions until power is cycled.

JDIFF Command

Command Type [General Operation and Configuration](#)

Description Specify or query the differential source of the receiver
Forces the system to go and use "diff" as the source.

Command Format Specify the differential mode
`$JDIFF,DIFF[,SAVE]<CR><LF>`

where:

- 'DIFF' (differential source) may be one of the following:

DIFF	Description
OTHER	Instruct the receiver to use external corrections input through the opposite port that is communicating
THIS	Instruct the receiver to use external corrections input through the same port that is communicating
PORTA or PORTB or PORTC or PORTD	Instruct the receiver to use external corrections input through the specified port
BEACON	Instruct the receiver to use RTCM corrections entering Port C at a fixed rate of 9600 baud. This input does not have to be from a beacon receiver, such as SBX. However, this is a common source of corrections.
WAAS	Instruct the receiver to use SBAS. This is also the response when running the local dif application as the base.
RTK	Response when running the local dif or rover RTK application for the rover.
LBAND	Instruct the receiver to use OmniSTAR.
X	Instruct the receiver to use e-Dif mode
NONE	Instruct the receiver to operate in autonomous mode

- 'SAVE' = optional field, saves the differential source into flash memory so that if you reset power the receiver will boot with the new differential source (it may take several seconds to save the differential source to flash memory)

Query the current DIFF setting

`$JDIFF<CR><LF>`

**Receiver
Response**

Receiver response when specifying the differential source

\$>

Receiver response when querying the differential source

\$>JDIFF,X

where 'X' is the differential source in the table above ('AUTO' is the response when queried in e-Dif)

Example

Issue the following command to query the receiver:

\$JDIFF<CR><LF>

...and if the differential source is WAAS, the response is:

\$>JDIFF,WAAS

**Additional
Information**

The status of this command is also output in the [JSHOW](#) message.

JDIFFX,EXCLUDE Command

Command Type [General Operation and Configuration](#)

Description Specify the differential sources to be excluded from operating in a multi-differential application

Command Format \$JDIFFX,EXCLUDE[,SBAS][,OMNIVBS][,OMNIHP][,RTCM2][,EDIF][,DFX][,CMR]
[,RTCM3][,ROX]<CR><LF>

Receiver Response \$>

Example Issue the following command to exclude RTCM3:

\$JDIFFX,EXCLUDE,RTCM3<CR><LF>

Additional Information

JDIFFX,GNSSOUT Command

Command Type [General Operation and Configuration](#)

Description Specify GNSS output in correction formats or query the current setting

Command Format Specify the GNSS output

\$JDIFFX,GNSSOUT,gnss,x<CR><LF>

where:

- gnss = GNSS type to output in correction formats (GPS and/or GLONASS)
- x = NO (do not output in correction formats) or YES (output in correction formats)

Query the current setting

Query what GNSS types are output in correction formats

\$JDIFFX,GNSSOUT<CR><LF

Query if a specific GNSS type is output in correction formats

\$JDIFFX,GNSSOUT,gnss<CR><LF

Receiver Response Receiver response when specifying the GNSS output

\$>

Receiver response when querying the current output setting

Query what GNSS types are output in correction formats

\$>JDIFFX,GNSSOUT<CR><LF

Query if a specific GNSS type is output in correction formats

\$>JDIFFX,GNSSOUT,gnss,x<CR><LF>

where 'x' is YES or NO

Example Query what GNSS types are output in correction formats

Command: \$JDIFFX,GNSSOUT<CR><LF>

Response if just GPS: \$>JDIFFX,GNSSOUT,GPS

Response if both GPS and GLONASS: \$>JDIFFX,GNSSOUT,GPS,GLONASS

Query if GLONASS is output in correction formats

Command: \$JDIFFX,GNSSOUT,GLONASS<CR><LF>

Response if GLONASS is not output: \$>JDIFFX,GNSSOUT,GLONASS,NO

Specify that GPS is output in correction formats

Command: \$JDIFFX,GNSSOUT,GPS,YES<CR><LF>

Response: \$>JDIFFX,GNSSOUT,GPS

**Additional
Information**

JDIFFX,INCLUDE Command

Command Type [General Operation and Configuration](#)

Description	Specify the differential sources to be allowed to operate in a multi-differential application
--------------------	---

Command Format	<code>\$JDIFFX,INCLUDE[,SBAS][,OMNIVBS][,OMNIHP][,RTCM2][,EDIF][,DFX][,CMR][,RTCM3][,ROX]<CR><LF></code>
-----------------------	--

Receiver Response	<code>\$></code>
--------------------------	---------------------

Example	Issue the following command to include CMR:
----------------	---

`$JDIFFX,INCLUDE,CMR<CR><LF>`

Additional Information	For example, if an Eclipse II receiver with SBAS, OmniSTAR, and RTK-base in the same application (multi-diff) has no active OmniSTAR subscription:
-------------------------------	--

1. The receiver tries XP/HP and when it is not found, falls back to VBS.
2. The receiver tries VBS and when it is not found, falls back to WAAS.
3. No warnings when subscription has expired – user expects a certain level of accuracy with OmniSTAR, not SBAS level accuracy.

If you do not actively watch the OmniSTAR end date, you could potentially use SBAS without knowing it. This command limits the diff sources to ensure a certain level of accuracy is retained.

JDIFFX,SOURCE Command

Command Type [General Operation and Configuration](#)

Description Query the receiver for the differential source

Command Format \$JDIFFX,SOURCE<CR><LF>

Receiver Response \$>JDIFFX,source
where 'source' is the differential source

Example Response if OmniSTAR is the differential source
\$>JDIFFX,SOURCE,LBAND

Response if RTK is the differential source through Port B
\$>JDIFFX,SOURCE,PORTB

Additional Information

JDIFFX,TYPE Command

Command Type [General Operation and Configuration](#)

Description Query the receiver for the differential type

Command Format \$JDIFFX,TYPE<CR><LF>

Receiver Response \$>JDIFFX,type
where 'type' is the differential type

Example Response if OmniSTAR HP (L2) is the differential type
\$>JDIFFX,TYPE,OMNIHP

Response if OmniSTAR VBS (L1) is the differential type
\$>JDIFFX,TYPE,OMNIVBS

Response if RTK (ROX) is the differential type
\$>JDIFFX,TYPE,ROX

Additional Information

JFLASH Command Overview

The JFLASH command is used to perform file operations via a USB flash drive on Eclipse and Eclipse II based receivers.

Command	Description
JFLASH,DIR	Display the files on a USB flash drive
JFLASH,FILE,CLOSE	Close an open file on a USB flash drive
JFLASH,FILE,NAME	Open a specific file, append to a specific file, or display the file name of the open file on a USB flash drive
JFLASH,FILE,OPEN	Create and open a file with an automatically generated file name on a USB flash drive
JFLASH,FREESPACE	Display the free space in kilobytes (KB) on a USB flash drive
JFLASH,NOTIFY,CONNECT	Enable/disable the automatic response when a USB flash drive is inserted or removed
JFLASH,QUERYCONNECT	Manually verify if a USB flash drive is connected or disconnected

JFLASH,DIR Command

Command Type [General Operation and Configuration](#)

Description Display the files on a USB flash drive

You can only display files at the root level of the flash drive (you cannot navigate into subdirectories).

Command Format \$JFLASH,DIR<CR><LF>

Receiver Response \$>JFLASH,file1
\$>JFLASH,file2
\$>JFLASH,file3
...
\$>JFLASH,filen

One line appears for each file at the root level of the flash drive.

Example If you issue the \$JFLASH,DIR command and the root level of the flash drive contains the following files:

hemi_1.bin, hemi_2.bin, hemi_3.bin

the response is:

\$>JFLASH,hemi_1.bin
\$>JFLASH,hemi_2.bin
\$>JFLASH,hemi_3.bin

Additional Information

JFLASH,FILE,CLOSE Command

Command Type [General Operation and Configuration](#)

Description Close an open file on a USB flash drive

Closing a file does not turn off the messages being written to the flash drive; it just closes the file so you can safely remove the flash drive.

Caution: Close the file before removing the flash drive. Failure to do so may corrupt the file.

Command Format `$JFLASH,FILE,CLOSE<CR><LF>`

Receiver Response `$>JFLASH,CLOSE mass_storage:0:\filename`

Example If you issue the `$JFLASH,FILE,CLOSE` command and the 'hemi_4.bin' file on the flash drive is currently open, the response is:

`$>JFLASH,CLOSE mass_storage:0:\HEMI_4.BIN`

Additional Information

JFLASH,FILE,NAME Command

Command Type [General Operation and Configuration](#)

Description Open a specific file, append to a specific file, or display the file name of the open file on a USB flash drive

Command Format Open a specific file (overwrite or append)

`$JFLASH, FILE, NAME, filename[, APPEND] <CR> <LF>`

where:

- 'filename' is the name of the file and it must be a legal 8.3 file name
- ',APPEND' is an optional field that allows you to append data to the file

Warning: Using this command without the ',Append' option overwrites the existing file without warning.

Display the name of the open file

`$JFLASH, FILE, NAME <CR> <LF>`

Receiver Response Response from issuing command to open an existing file or append to an existing file

`$>JFLASH, OPEN mass_storage:0:\filename`

Response from issuing command to display the name of the open file

`$>JFLASH, mass_storage:0:\filename`

If you attempt to display the name of the open file and no file is actually open the response is:

`$>JFLASH, NO FILE OPEN`

Example If you issue the following command to open file hemi_4.bin on a USB flash drive:

`$JFLASH, FILE, NAME, hemi_4.bin <CR> <LF>`

the response is:

`$>JFLASH, mass_storage:0:\HEMI_4.BIN`

Additional Information

JFLASH,FILE,OPEN Command

Command Type [General Operation and Configuration](#)

Description Create and open a file with an automatically generated file name (hemi_1.bin ... hemi_99.bin) on a USB flash drive (only 8.3 file format is allowed)

Command Format \$JFLASH,FILE,OPEN<CR><LF>

Receiver Response \$>JFLASH,OPEN mass_storage:0:\filename
where 'filename' is the name of the new file

Example If you issue the \$JFLASH,FILE,OPEN command and the root level of the flash drive contains the following files:

hemi_1.bin, hemi_2.bin, hemi_3.bin

the response is:

\$>JFLASH,OPEN mass_storage:0:\HEMI_4.bin

Additional Information

JFLASH,FREESPACE Command

Command Type [General Operation and Configuration](#)

Description Display the free space in kilobytes (KB) on a USB flash drive

You can use a flash drive larger than 4GB; however, this command will not display a number greater than 4GB.

Command Format \$JFLASH,FREESPACE<CR><LF>

Receiver Response \$>JFLASH,FREESPACE, numbytes bytes

where 'numbytes' is the number of kilobytes

Example The following response indicates a USB flash drive with approximately 2GB of free space.

\$>JFLASH,FREESPACE, 2001731584 bytes

Additional Information

JFLASH,NOTIFY,CONNECT Command

Command Type [General Operation and Configuration](#)

Description Enable/disable the automatic response when a USB flash drive is inserted or removed (if port is not specified the response will be sent to the port that issued the command)

Command Format \$JFLASH,NOTIFY,CONNECT,R[,PORT]<CR><LF>

where:

- 'R' is the message status variable (0 = Off, 1 = On)
- ',PORT' is an optional field you use to specify the port to which the response will be sent (if you do not specify a port, the response is sent to the port from which you issued the command)

Receiver Response Response to issuing command to enable notification

\$>

Response to inserting a flash drive if notification is enabled

\$>JFLASH,CONNECTED

Response to removing a flash drive if notification is enabled

\$>JFLASH,DISCONNECTED

Additional Information

JFLASH,QUERYCONNECT Command

Command Type [General Operation and Configuration](#)

Description Manually verify if a USB flash drive is connected or disconnected

Command Format \$JFLASH, QUERYCONNECT<CR><LF>

Receiver Response Response to verifying the connection status of a flash drive if the flash drive is connected

\$>JFLASH, CONNECTED
\$>

Response to verifying the connection status of a flash drive if the flash drive is disconnected

\$>JFLASH, DISCONNECTED
\$>

Additional Information

JFREQ Command

Command Type [OmniSTAR](#)

Description Tune the OmniSTAR receiver (manually or automatically) or query the receiver for the current setting

Command Format Tune the OmniSTAR receiver

To manually tune the receiver:

```
$JFREQ, freq, symb<CR><LF>
```

where:

- 'freq' is the frequency in kHz (reply is in MHz)
- 'symb' is the symbol baud rate (1200 or 2400)

To auto-tune the receiver:

```
$JFREQ, 0<CR><LF>
```

Note: You must restart the OmniSTAR receiver (either by cycling power to the OmniSTAR receiver or by issuing the [JBOOT.OMNI](#) command) for changes to take effect.

Query the current setting

```
$JFREQ<CR><LF>
```

Receiver Response Response to issuing command to tune receiver

```
$>
```

Response to querying the current setting

```
$>JLBEAM, Sent sfreq, Used ufreq, Baud baud, Geo lon[, AUTO]
```

where:

Response Component	Description
sfreq	Frequency to which the OmniSTAR receiver is instructed to tune (in this example, 1557.8550 MHz)
ufreq	Frequency to which the OmniSTAR receiver is tuned
baud	Baud Rate of the signals being received
lon	Approximate longitude of the geostationary satellite to which the OmniSTAR receiver is tuned
AUTO	[Optional Field] 'AUTO' appears at the end of the query response only when the OmniSTAR receiver is in 'auto-tune' mode.

Example Manually Tune a Frequency (command and response)

```
$JFREQ,1557835,1200
$>
```

Auto-Tune a Frequency based on Geographic Location (command and response)

```
$JFREQ,0
$>
```

Query a Manually Tuned Receiver (response)

```
$>JLBEAM,Sent 1557.8350,Used 1557.8350,Baud 1200,Geo -101
```

Query an Auto-Tuned Receiver (response)

```
$>JLBEAM,Sent 1557.8550,Used 1557.8550,Baud 1200,Geo -101,AUTO
```

**Additional
Information**

The status of this command is also output when issuing the [JSHOW](#) command.

The following table provides frequency information for the OmniSTAR satellites. This information is subject to change. Visit www.omnistar.com for up-to-date information.

Coverage Area	Longitude	Frequency	Baud Rate	Satellite Name
Eastern U.S.	101 West	1557.8450	1200	MSV-E
Central U.S.	101 West	1557.8350	1200	MSV-C
Western U.S.	101 West	1557.8550	1200	MSV-W
North, Central, and South America, including the Caribbean	98 West	1535.1375	1200	AM-SAT
Asia, Pacific Islands	109 East	1535.1375	1200	AP-SAT
Europe, Africa, Middle East	25 East	1537.440	1200	EUSAT
Australia, Far East	160 East	1535.185	1200	OCSAT

If you are already locked onto an OmniSTAR signal, you will need to break lock on the OmniSTAR satellite before JFREQ will manually tune to your new signal. To do this, either disconnect the antenna momentarily, cycling power to the receiver, issuing the JBOOT,OMNI command, or block signal to the antenna physically, for example by covering it with something metallic.

JGEO Command

Command Type [SBAS](#)

Description Display information related to the current frequency of SBAS and its location in relation to the receiver's antenna

Command Format \$JGEO[, ALL] <CR><LF>

where 'ALL' is an optional field that displays information for all SBAS satellites (including those not being used)

Receiver Response \$>JGEO , SENT=1575.4200 , USED=1575.4200 , PRN=prn , LON=lon , EL=ele , AZ=az

where:

Response Component	Description
JGEO	Message header
Sent=1575.4200	Frequency sent to the digital signal processor
Used=1575.4200	Frequency currently used by the digital signal processor
PRN=prn	WAAS satellite PRN number
Lon=-lon	Longitude of the satellite
El=ele	Elevation angle from the receiver antenna to the WAAS satellite, reference to the horizon
AZ=az	Azimuth from the receiver antenna to the WAAS satellite, reference to the horizon

Example To display information related to the current frequency of SBAS issue the following command:

```
$JGEO[ , ALL ] <CR><LF>
```

The response is then:

```
$>JGEO , SENT=1575.4200 , USED=1575.4200 , PRN=122 , LON=-54 , EL=9.7 , AZ=114.0
```

To display information for dual SBAS satellites issue the following command:

```
$JGEO[ , ALL ] <CR><LF>
```

The response is:

```
$>JGEO , SENT=1575.4200 , USED=1575.4200 , PRN=122 , LON=-54 , EL=9.7 , AZ=114.0
$>JGEO , SENT=1575.4200 , USED=1575.4200 , PRN=134 , LON=178 , EL=5.0 , AZ=252.6
```

The first line of output is identical to the output from the first JGEO query above; however, the second line of output provides information on the WAAS satellite not being currently used. Both lines of output follow the same format.

Additional Information

JHP Command Overview

The JHP command is used to control the operation of the OmniSTAR HP/XP engine.

Command	Description
JHP,LIMIT	Specify the OmniSTAR HP/XP convergence threshold (range is 0.0 to 1.0 m) or display the threshold as compared to the RMS value in the GPGST message
JHP,MODE,AUTOSEED	Enable or disable the AUTOSEED feature when operating in LBAND mode and using OmniSTAR XP/HP service, or query the current setting
JHP,MODE,IGNORECONVERGE	If using the JHP,LIMIT command to specify the OmniSTAR HP/XP convergence threshold, use this command to set the receiver to ignore when the OmniSTAR engine indicates it is converged or query the current setting
JHP,POS	Query the receiver for the stored position with standard deviations (StDevs) to be used with the JHP,SEED command
JHP,POS,LAT,LON,HGT	Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the JHP,SEED command
JHP,POS,LAT,LON,HGT,...,OTHER	(For use with AUTOSEED feature) Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the JHP,MODE,AUTOSEED command
JHP,POS,OTHER	(For use with AUTOSEED feature) Query the receiver for the stored position with standard deviations (StDevs) to be used with the JHP,MODE,AUTOSEED command
JHP,POS,PRESENT	Save the current location (lat, lon, height) and standard deviations into non-volatile memory (provided the sum of the location standard deviations (StDev) < 0.6 m) to be used with the JHP,SEED command
JHP,RESET,ACCURACY	Reset the HP convergence by forcing the solution to the current location but with very large standard deviations
JHP,RESET,ENGINE	Reset the HP engine, forcing the solution to converge (this will also force an AUTOSEED location to reconverge)
JHP,SEED	Initialize the OmniSTAR HP algorithm with the saved position and saved standard deviations
JHP,SEED,LAT,LON,HGT	Initialize the OmniSTAR HP algorithm with the given coordinates and optional standard deviations (this command has the combined effect of the JHP,POS,LAT,LON,HGT and JHP,SEED commands)
JHP,STATIC	Place the OmniSTAR HP engine into or out of static mode, or query the current setting
JHP,STATUS,AUTOSEED	Displays the status of the AUTOSEED initialization progress

JHP,LIMIT Command

Command Type [OmniSTAR](#)

Description Specify the OmniSTAR HP/XP convergence threshold (range is 0.0 to 1.0 m) or display the threshold as compared to the RMS value in the [GPGST](#) message

Command Format Set the convergence threshold

`$JHP,LIMIT,thresh<CR><LF>`

where "thresh" is the threshold range of 0.000 to 1.000 m

Query the current setting (display convergence threshold)

`$JHP,LIMIT<CR><LF>`

Receiver Response Response to issuing command to set convergence threshold

`$>`

Response to querying the current setting

`$>JHP,LIMIT,thresh`

Example Issue the following command to specify a threshold range of 0.5 m:

`$JHP,LIMIT,0.5<CR><LF>`

If you then issue `$JHP,LIMIT<CR><LF>` to query the receiver for the current threshold, the following will appear:

`$>JHP,LIMIT,0.500`

Additional Information Use this to set a convergence threshold for defining when a [GPGGA](#) message will indicate a quality indicator as fixed (quality indicator 4) rather than float (quality indicator 5). If you do not set a limit, the receiver will use the limit defined in the OmniSTAR engine.

When setting a convergence threshold, use the [JHP.MODE.IGNORECONV](#) command to turn off the automatically defined limit in the OmniSTAR engine.

JHP,MODE,AUTOSEED Command

Command Type [OmniSTAR](#)

Description Enable or disable the AUTOSEED feature when operating in LBAND mode and using OmniSTAR XP/HP service, or query the current setting

Warning! Hemisphere GPS does not recommend using the AUTOSEED feature when there is risk of the antenna moving more than 2.5 cm (1 in) while the system is powered off.

Command Format Enable/disable AUTOSEED feature

To enable the AUTOSEED feature:

```
$JHP,MODE,AUTOSEED,YES<CR><LF>
```

To disable the AUTOSEED feature:

```
$JHP,MODE,AUTOSEED,NO<CR><LF>
```

Query the current setting

```
$JHP,MODE,AUTOSEED<CR><LF>
```

Receiver Response Response to issuing command to enable/disable AUTOSEED feature

```
$>
```

Response to querying the current setting

If setting is currently enabled the response is:

```
$>JHP,MODE,AUTOSEED,YES
```

If setting is currently disabled the response is:

```
$>JHP,MODE,AUTOSEED,NO
```

Additional Information [JHP,POS,PRESENT](#), [JHP,POS,LAT,LON,HGT](#), [JHP,STATUS,AUTOSEED](#)

JHP,MODE,IGNORECONV Command

Command Type [OmniSTAR](#)

Description If using the [JHP,LIMIT](#) command to specify the OmniSTAR HP/XP convergence threshold, use this command to set the receiver to ignore when the OmniSTAR engine indicates it is converged (the threshold setting stored with the JHP,LIMIT command is then used as the only determining criteria for indicating that the solution has converged to the defined accuracy) or query the current setting

Command Format Ignore/accept OmniSTAR engine convergence indication

To ignore OmniSTAR engine convergence indication:

```
$JHP,MODE,IGNORECONV,YES<CR><LF>
```

To accept OmniSTAR engine convergence indication:

```
$JHP,MODE,IGNORECONV,NO<CR><LF>
```

Query the current setting

```
$JHP,MODE,IGNORECONV<CR><LF>
```

Receiver Response Response to issuing command to ignore/accept OmniSTAR engine convergence indication

```
$>
```

Response to querying the current setting

If currently set to ignore the response is:

```
$>JHP,MODE,IGNORECONV,YES
```

If currently set to accept the response is:

```
$>JHP,MODE,IGNORECONV,NO
```

Additional Information If you set a convergence limit using the [JHP,LIMIT](#) command, using JHP,MODE,IGNORECONV,NO will require BOTH the predefined OmniSTAR convergence limit AND the value entered in JHP,LIMIT to be met before the [GPGGA](#) message will indicate a quality indicator of 4, meaning “converged”.

JHP,POS Command

Command Type [OmniSTAR](#)

Description Query the receiver for the stored position with standard deviations (StDevs) to be used with the [JHP,SEED](#) command

Command Format \$JHP , POS<CR><LF>

Receiver Response Response if there is a saved position
\$>JHP , POS , LAT , LON , HEIGHT , STDEV

Response if there is no saved position
\$>JHP , POS , LAT , LON , HEIGHT

Example If you issue the command and there is a saved position, the response will be similar to the following:
\$>JHP , POS ,

Additional Information [JHP,SEED](#)

JHP,POS,LAT,LON,HGT Command

Command Type [OmniSTAR](#)

Description Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the [JHP,SEED](#) command

Command Format \$JHP,POS,lat,lon,hgt[,latstdev,lonstdev,hgtstdev]<CR><LF>

where

Command Component	Description
lat	Latitude in decimal degrees
lon	Longitude in decimal degrees
hgt	<p>You must enter HEIGHT as ellipsoidal height in meters. Ellipsoidal height can be calculated by adding the altitude and the geoidal separation, both available from the GPGGA message.</p> <p>Example:</p> <p>\$GPGGA,173309.00,5101.04028,N,11402.38289,W,2,07,1.4,1071.0, M,-17.8,M,6.0,0122*48 ellipsoidal height = 1071.0 + (-17.8) = 1053.2 meters</p>
latstdev	<p><i>Optional field</i></p> <p>Standard deviation of latitude in meters</p>
lonstdev	<p><i>Optional field</i></p> <p>Standard deviation of longitude in meters</p>
hgtstdev	<p><i>Optional field</i></p> <p>Standard deviation of height in meters</p>

Receiver Response \$>

Additional Information [JHP,SEED](#)

JHP,POS,LAT,LON,HGT,,,,OTHER Command

Command Type [OmniSTAR](#)

Description For use with AUTOSEED feature

Save lat, lon, hgt and optionally save corresponding standard deviations into non-volatile memory, to be used with the [JHP,MODE,AUTOSEED](#) command

Command Format \$JHP,POS,lat,lon,hgt[,latstdev,lonstdev,hgtstdev],OTHER<CR><LF>

where

Command Component	Description
lat	Latitude in decimal degrees
lon	Longitude in decimal degrees
hgt	<p>You must enter HEIGHT as ellipsoidal height in meters. Ellipsoidal height can be calculated by adding the altitude and the geoidal separation, both available from the GPGGA message.</p> <p>Example:</p> <p>\$GPGGA,173309.00,5101.04028,N,11402.38289,W,2,07,1.4,1071.0, M,-17.8,M,6.0,0122*48 ellipsoidal height = 1071.0 + (-17.8) = 1053.2 meters</p>
latstdev	<i>Optional field</i> Standard deviation of latitude in meters
lonstdev	<i>Optional field</i> Standard deviation of longitude in meters
hgtstdev	<i>Optional field</i> Standard deviation of height in meters

Receiver Response \$>

Additional Information [JHP,MODE,AUTOSEED](#), [JHP,POS,OTHER](#)

JHP,POS,OTHER Command

Command Type [OmniSTAR](#)

Description For use with AUTOSEED feature

Query the receiver for the stored position with standard deviations (StDevs) to be used with the [JHP,MODE,AUTOSEED](#) command

Command Format \$JHP , POS , OTHER<CR><LF>

Receiver Response

Additional Information [JHP,MODE,AUTOSEED](#)

JHP,POS,PRESENT Command

Command Type [OmniSTAR](#)

Description Save the current location (lat, lon, height) and standard deviations into non-volatile memory (provided the sum of the location standard deviations (StDev) < 0.6 m) to be used with the [JHP,SEED](#) command

Command Format \$JHP , POS , PRESENT<CR><LF>

Receiver Response \$>

Additional Information [JHP,SEED](#)

JHP,RESET,ACCURACY Command

Command Type [OmniSTAR](#)

Description Reset the HP convergence by forcing the solution to the current location but with very large standard deviations

Command Format \$JHP , RESET , ACCURACY<CR><LF>

Receiver Response \$>

Additional Information You can watch the [GPGGA](#) message quality indicator and the [GPGST](#) message estimated accuracy values to see the effect of resetting the OmniSTAR HP accuracy.

JHP,RESET,ENGINE Command

Command Type [OmniSTAR](#)

Description Reset the HP engine, forcing the solution to converge (this will also force an AUTOSEED location to reconverge)

Command Format \$JHP , RESET , ENGINE<CR><LF>

Receiver Response \$>

Additional Information You can watch the [GPGGA](#) message quality indicator and the [GPGST](#) message estimated accuracy values to see the effect of resetting the OmniSTAR HP engine.

JHP,SEED Command

Command Type [OmniSTAR](#)

Description Initialize the OmniSTAR HP algorithm with the saved position and saved standard deviations

- To enter a saved position, see the [JHP,POS,LAT,LON,HGT](#) and [JHP,POS,PRESENT](#) commands
- Alternately, use the ,OTHER option to seed using the location stored for use with the [JHP,MODE,AUTOSEED](#) feature

Command Format \$JHP,SEED[,OTHER]<CR><LF>

Receiver Response \$>

If the coordinates to which you are attempting to initialize are not close enough to your current location, the response is:

\$>JHP,SEED,Current Position Too Far From Seed

Additional Information [JHP,POS,LAT,LON,HGT](#), [JHP,POS,PRESENT](#)

JHP,SEED,LAT,LON,HGT Command

Command Type [OmniSTAR](#)

Description Initialize the OmniSTAR HP algorithm with the given coordinates and optional standard deviations (this command has the combined effect of the [JHP,POS,LAT,LON,HGT](#) and [JHP,SEED](#) commands)

Command Format \$JHP,SEED,LAT,LON,HGT[,LatStDev,LonStDev,HgtStDev]

Receiver Response \$>

If the coordinates to which you are attempting to initialize are not close enough to your current location, the response is:

\$>JHP,SEED,Current Position Too Far From Seed

Additional Information [JHP,POS,LAT,LON,HGT](#), [JHP,SEED](#)

JHP,STATIC Command

Command Type [OmniSTAR](#)

Description	Place the OmniSTAR HP engine into or out of static mode, or query the current setting
--------------------	---

Command Format	Place HP into or out of static mode
-----------------------	-------------------------------------

To place HP into static mode:

```
$JHP , STATIC , YES<CR><LF>
```

To place HP out of static mode:

```
$JHP , STATIC , NO<CR><LF>
```

Query the current setting

```
$JHP , STATIC<CR><LF>
```

Receiver Response	Response to issuing command to place HP into or out of static mode
--------------------------	--

```
$>
```

Response to querying the current setting
--

If HP is currently in static mode the response is:

```
$>JHP , STATIC , YES
```

If HP is currently not in static mode the response is:

```
$>JHP , STATIC , NO
```

Additional Information	If the receiver detects motion while in static mode, it changes its status to effectively the same as issuing the \$JHP,STATIC,NO command.
-------------------------------	--

JHP,STATUS,AUTOSEED Command

Command Type [OmniSTAR](#)

Description Displays the status of the AUTOSEED initialization progress

Command Format \$JHP,STATUS,AUTOSEED<CR><LF>

Receiver Response \$>JHP,STATUS,AUTOSEED,status

Where 'status' is one of following:

- NoHP
- Disabled
- Seeding
- Failed_NoSeed
- Failed_Moved
- Failed_Timeout
- Success

Additional Information You can watch the [GPGGA](#) message quality indicator and the [GPGST](#) message estimated accuracy values to see the effect of resetting the OmniSTAR HP accuracy.

JI Command

Command Type [General Operation and Configuration](#)

Description Display receiver information, such as its serial number and firmware version

Command Format \$JI<CR><LF>

Receiver Response \$>JI,SN,FLT,HW,PROD,SDATE,EDATE,SW,DSP<CR><LF>

where:

Response Component	Description
SN	Serial number of the GPS engine
FLT	Fleet number
HW	Hardware version
PROD	Production date code
SDATE	Subscription begin date when running OmniSTAR application; not applicable when running all other applications
EDATE	Subscription expiration date when running OmniSTAR application OR receiver subscription code when running all other applications (see Interpreting the \$JI and \$JK 'Date'/Subscription Codes to determine the meaning of the subscription code)
SW	Application software version number
DSP	DSP version (only valid for OmniSTAR applications)

Example From a Crescent Vector:

\$>JI,452204,1,7,02122009,01/01/1900,01/01/3007,1.5Pa,46

From a Crescent with OmniSTAR:

\$>JI,883765,1,7,12052010,01/06/1980,06/30/2011,4.9Pa,11

Additional Information

JK Command

Command Type [General Operation and Configuration](#)

Description Subscribe the receiver to various options, such as higher update rates, e-Dif (or base station capability) or L-Dif

or

Query for the current subscription expiration date when running OmniSTAR application or the receiver subscription code when running all other applications

Command Format Subscribe the receiver to specific options

`$JK , X...<CR><LF>`

where 'X...' is the subscription key provided by Hemisphere GPS and is 10 characters in length

Query the current setting

`$JK<CR><LF>`

Receiver Response Response to issuing command to subscribe

`$>`

Response to querying the current setting when running OmniSTAR applications

`$>JK,EndDate,1HzOnly`

where:

- 'EndDate' is the subscription end date
- '1HzOnly' has a value of 1 if the receiver is limited to 1 Hz output (if the receiver is subscribed to a minimum of 10 Hz output this field is omitted)

Response to querying the current setting when running all other applications

`$>JK,SubscriptionCode,1HzOnly`

where:

- 'SubscriptionCode' is the subscription code (see [Interpreting the \\$JI and \\$JK 'Date'/Subscription Codes](#) to determine the meaning of the subscription code)
- '1HzOnly' has a value of 1 if the receiver is limited to 1 Hz output (if the receiver is subscribed to a minimum of 10 Hz output this field is omitted)

Example If you query the receiver for the current setting when running OmniSTAR applications the response will appear similar to the following:

\$>JK,06/30/2011,0

If you query the receiver for the current setting when running any other application the response will appear similar to the following (Crescent Vector example response shown):

\$>JK,01/01/3007,7

**Additional
Information**

JLBEAM Command

Command Type [OmniSTAR](#)

Description Display the information of each spot beam currently in use by the OmniSTAR receiver

Command Format \$JLBEAM<CR><LF>

Receiver Response \$>JLBEAM,Sent freq,Used freq,Baud xxx,Geo xxx
 (1)
 \$>JLBEAM,freq1,lon1,lat1,baud1,satlon1
 (2)
 .
 .
 .
 \$>JLBEAM,freqn,lonn,latn,baudn,satlonn

where:

Response Component	Description
"Sent" freq	Frequency sent to the digital signal processor (DSP)
"Used" freq	Frequency currently being used by the digital signal processor (DSP)
"Baud" xxx	Currently used baud rate of the acquired signal
"Geo" xxx	Currently used satellites longitude (in degrees)

The output second line components are described in the following table:

Response Component	Description
freq	Frequency of the spot beam
lon	Longitude of the center of the spot beam (in degrees)
lat	Latitude of the center of the spot beam (in degrees)
baud	Baud rate at which this spot beam is modulated
satlon	Satellites longitude (in degrees)

Example \$>JLBEAM,Sent 1551.4890,Used 1551.4890,Baud 1200,Geo -101
 \$>JLBEAM,1556.8250,-88,45,1200,(-101)
 \$>JLBEAM,1554.4970,-98,45,1200,(-101)
 \$>JLBEAM,1551.4890,-108,45,1200,(-101)
 \$>JLBEAM,1531.2300,25,50,1200,(16)
 \$>JLBEAM,1535.1375,-75,0,1200,(-98)

```
$>JLBEAM,1535.1375,-165,13,1200,(-98)
$>JLBEAM,1535.1525,20,6,1200,(25)
$>JLBEAM,1558.5100,135,-30,1200,(160)
$>JLBEAM,1535.1375,90,15,1200,(109)
$>JLBEAM,1535.1375,179,15,1200,(109)
```

Additional Information

JLIMIT Command

Command Type [General Operation and Configuration](#)

Description Set the threshold of estimated horizontal performance for which the DGPS position LED is illuminated or query the current setting

Command Format Set the threshold of estimated horizontal performance

```
$JLIMIT, LIMIT<CR><LF>
```

where 'LIMIT' is the new limit in meters

Query the current setting

```
$JLIMIT<CR><LF>
```

Receiver Response Receiver response when setting the threshold of estimated horizontal performance

```
$>
```

Receiver response when querying the current threshold of estimated horizontal performance

```
$>JLIM, RESID, LIMIT
```

where 'LIMIT' is the limit in meters

Example To set the threshold to 5 m issue the following command:

```
$JLIMIT, 5<CR><LF>
```

If you then query the receiver with `$JLIMIT<CR><LF>` the response is:

```
$JLIM, RESID, 5.00
```

Additional Information The default value for this parameter is a conservative 10.00 m.
The status of this command is also output in the [JSHOW](#) message.

JLXBEAM Command

Command Type [OmniSTAR](#)

Description Display spot beam debug information

Command Format \$JLXBEAM<CR><LF>

Receiver Response \$>JLBEAMEX
 \$> Beam:1,DDSfreq1,symbol1,lon1,lat1,lonrad1,latrad1,beamrot1,satlon1,*
 \$> Beam:2,DDSfreq2,symbol2,lon2,lat2,lonrad2,latrad2,beamrot2,satlon2,*
 .
 .
 \$> Beam:n,DDSfreqn,symboln,lonn,latn,lonradn,latradn,beamrotn,satlonn,*

where:

Response Component	Description
DDSfreq	DDS frequency
symbol	Symbol rate used for that particular spot beam
lon	Longitude of the spot beam centroid
lat	Latitude of the spot beam centroid
lonrad	Longitude radius of the spot beam
latrad	Latitude radius of the spot beam
beamrot	Rotation angle of the spot beam
satlon	Longitude of the L-band satellite
*	Reserved

Example \$>JLBEAMEX
 \$> Beam:22,1535125000,600,-26,40,2,41,0,9999,*
 \$> Beam:21,1535157500,600,65,30,31,18,-21,64,*
 \$> Beam:13,1535185000,1200,136,-25,23,28,-40,144,*
 \$> Beam:13,1535185000,1200,172,-40,13,26,-26,144,*
 \$> Beam:24,1557835000,1200,-100,49,6,28,0,-101,*
 \$> Beam:24,1557835000,1200,-101,66,12,6,0,-101,*
 \$> Beam:25,1557845000,1200,-74,52,12,30,-30,-101,*
 \$> Beam:26,1557855000,1200,-122,45,11,30,25,-101,*
 \$> Beam:8,1535137500,1200,-85,2,30,20,-5,-98,*
 \$> Beam:8,1535137500,1200,-60,-25,34,36,-20,-98,*
 \$> Beam:4,1535137500,1200,109,2,14,19,-27,109,*

```
$> Beam:4,1535137500,1200,140,38,27,51,-56,109,*  
$> Beam:7,1537440000,1200,23,-2,29,49,50,25,*  
$> Beam:7,1537440000,1200,14,59,41,23,34,25,*  
$> Beam:7,1537440000,1200,11,28,17,24,0,25,*
```

Additional Information

JMASK Command

Command Type [GPS](#)

Description Specify the elevation cutoff mask angle for the GPS engine

Any satellites below this mask angle will be ignored even if available. The default angle is 5° because satellites available below this angle will have significant tropospheric refraction errors.

Command Format \$JMASK , E<CR><LF>

where the elevation mask cutoff angle 'E' may be a value from 0 to 60°

Receiver Response \$>

Example To specify the elevation cutoff mask angle to 10° issue the following command:

\$JMASK , 10<CR><LF>

Additional Information To query the receiver for the current setting, issue the [JSHOW](#) command.

JMODE Overview

The JMODE command is used to control various GPS tracking parameters.

Command	Description
JMODE	Query receiver for status of JMODE settings
JMODE.FOREST	Turn the higher gain functionality (for tracking under canopy) on/off or query the current setting
JMODE.GPSONLY	Set the receiver to use GPS data in the solution or query the current setting (if GLONASS is available, setting to YES will cause the receiver to only use GPS data)
JMODE.L1ONLY	Set the receiver to use L1 data even if L2 data is available or query the current setting
JMODE.MIXED	Include satellites that do not have DGPS or SBAS corrections in the solution
JMODE.NULLNMEA	Enable/disable output of NULL fields in NMEA 0183 messages when no there is no fix (when position is lost)
JMODE.SBASR	Enable/disable SBAS ranging
JMODE.TIMEKEEP	Enable/disable continuous time updating in NMEA 0183 messages when there is no fix (when position is lost)
JMODE.TUNNEL	Enable/disable faster reacquisition after coming out of a tunnel or query the current setting

JMODE Command

Command Type [General Operation and Configuration](#)

Description Query receiver for status of JMODE settings

Command Format \$JMODE<CR><LF>

Receiver Response \$>JMODES[,BASE][,FOREST][,GPSONLY][,L1ONLY][,MIXED][,NULLNMEA][,SBASR][,TIMEKEEP][,TUNNEL][,WIDESEARCH]

Example If FOREST and TUNNEL are set to ON and all others (MIXED, NULLNMEA, SBASR, and TIMEKEEP) are set to OFF and you issue the JMODE command the receiver response will be:

\$JMODES , TUNNEL , FOREST

If all features are set to OFF and you issue the JMODE command the receiver response will be:

\$JMODES

Additional Information The status of this command is also output in the [JSHOW](#) response. For example, if TUNNEL is set to ON and all other JMODE options are set to OFF then the JMODE part of the response to the JSHOW command is as follows:

\$>JSHOW , MODES , TUNNEL

JMODE,FOREST Command

Command Type [General Operation and Configuration](#)

Description Turn the higher gain functionality (for tracking under canopy) on/off or query the current setting.

This command is useful if you are trying to maximize the likelihood of calculating a position, but are willing to sacrifice accuracy. See also [JMODE.MIXED](#).

Command Format Turn enable/disable high gain functionality

To enable high gain functionality:

```
$JMODE,FOREST,YES<CR><LF>
```

To disable high gain functionality:

```
$JMODE,FOREST,NO<CR><LF>
```

Query the current setting

```
$JMODE,FOREST<CR><LF>
```

Receiver Response Response to issuing command to turn functionality on/off

```
$>
```

Response to querying the current setting

If high gain functionality is currently enabled the response is:

```
$>JMODE,FOREST,ON
```

If high gain functionality is currently disabled the response is:

```
$>JMODE,FOREST,OFF
```

Additional Information

JMODE,GPSONLY Command

Command Type [General Operation and Configuration](#)

Description Set the receiver to use GPS data in the solution or query the current setting (if GLONASS is available, setting to YES will cause the receiver to only use GPS data)

Command Format Enable/disable GPS-only operation

Enable GPS-only operation:

\$JMODE , GPSONLY , YES<CR><LF>

Disable GPS-only operation (use GLONASS as well if available):

\$JMODE , GPSONLY , NO<CR><LF>

Query the current setting

\$JMODE , GPSONLY<CR><LF>

Receiver Response Response to issuing command to turn enable/disable GPS-only operation

\$>

Response to querying the current setting

If GPS-only operation is currently enabled the response is:

\$>JMODE , GPSONLY , YES

If GPS-only operation is currently disabled the response is:

\$>JMODE , GPSONLY , NO

Additional Information

JMODE,L1ONLY Command

Command Type [General Operation and Configuration](#)

Description Set the receiver to use L1 data even if L2 data is available or query the current setting:

- When set to YES receiver will use OmniSTAR VBS or L1 RTK
- When set to NO receiver will use OmniSTAR XP/HP or L1/L2 RTK

Command Format Set receiver to use/not use L1 data even if L2 data is available

To use L1 data (even if L2 data is available):

```
$JMODE , L1ONLY , YES <CR> <LF>
```

To use L2 data if it is available:

```
$JMODE , L1ONLY , NO <CR> <LF>
```

Query the current setting

```
$JMODE , L1ONLY <CR> <LF>
```

Receiver Response Response to issuing command to turn functionality on/off

```
$>
```

Response to querying the current setting

If the receiver is currently using L1 data only even if L2 data is available the response is:

```
$>JMODE , L1ONLY , YES
```

If the receiver is currently using L2 data if it is available the response is:

```
$>JMODE , L1ONLY , NO
```

Additional Information

JMODE,MIXED Command

Command Type [General Operation and Configuration](#)

Description Include satellites that do not have DGPS or SBAS corrections in the solution

This command is useful if you are trying to maximize the likelihood of calculating a position, but are willing to sacrifice accuracy. See also [JMODE,FOREST](#).

Command Format To include/exclude satellites without DGPS or SBAS corrections

To include satellites without DGPS or SBAS corrections:

```
$JMODE,MIXED,YES<CR><LF>
```

To exclude satellites without DGPS or SBAS corrections:

```
$JMODE,MIXED,NO<CR><LF>
```

Query the current setting

```
$JMODE,MIXED<CR><LF>
```

Receiver Response Response to issuing command to include/exclude satellites without DGPS or SBAS corrections

```
$>
```

Response to querying the current setting

If satellites without differential corrections are currently included the response is:

```
$>JMODE,MIXED,ON
```

If satellites without differential corrections are currently excluded the response is:

```
$>JMODE,MIXED,OFF
```

Additional Information

JMODE,NULLNMEA Command

Command Type [General Operation and Configuration](#)

Description Enable/disable output of NULL fields in NMEA 0183 messages when no there is no fix (when position is lost)

This only applies to position portion of the messages; it does not affect the time portion of the message. If this setting is disabled and position is lost then the positioning parameters of the message from the most recent known position are repeated (instead of being NULL if enabled).

Command Format Enable/disable output of NULL fields in NMEA 0183 messages

To enable output:

```
$JMODE , NULLNMEA , YES<CR><LF>
```

To disable output:

```
$JMODE , NULLNMEA , NO<CR><LF>
```

Query the current setting

```
$JMODE , NULLNMEA<CR><LF>
```

Receiver Response Response to issuing command to enable/disable output of NULL fields in NMEA 0183 messages

```
$>
```

Response to querying the current setting

If setting is currently enabled the response is:

```
$>JMODE , NULLNMEA , ON
```

If setting is currently disabled the response is:

```
$>JMODE , NULLNMEA , OFF
```

Example If the most recent GPGLA message is as follows:

```
$GPGLA , 220715.00 , 3333.4254353 , N , 11153.3506065 , W , 2 , 10 , 1.0 , 406.614 , M , -  
26.294 , M , 6.0 , 1001*70
```

...and then position is lost and JMODE,NULLNMEA is set to OFF the GPGLA message repeats as follows (most recent known values do not change):

```
$GPGLA , 220715.00 , 3333.4254353 , N , 11153.3506065 , W , 2 , 10 , 1.0 , 406.614 , M , -  
26.294 , M , 6.0 , 1001*70
```

For the same message, if position is lost and JMODE,NULLNMEA is set to ON the GPGLA message repeats as follows (position parameters are NULL):

```
$GPGLA , 220716.00 , , , , , 0 , , , , M , , M , , *48
```

Additional Information

JMODE,SBASR Command

Command Type [General Operation and Configuration](#)

Description Enable/disable SBAS ranging

Command Format Enable/disable SBAS ranging

To enable SBAS ranging:

```
$JMODE , SBASR , YES<CR><LF>
```

To disable SBAS ranging:

```
$JMODE , SBASR , NO<CR><LF>
```

Query the current setting

```
$JMODE , SBASR<CR><LF>
```

Receiver Response Response to issuing command to enable/disable SBAS ranging

```
$>
```

Response to querying the current setting

If setting is currently enabled the response is:

```
$>JMODE , SBASR , ON
```

If setting is currently disabled the response is:

```
$>JMODE , SBASR , OFF
```

Additional Information

JMODE,TIMEKEEP Command

Command Type [General Operation and Configuration](#)

Description Enable/disable continuous time updating in NMEA 0183 messages when there is no fix (when position is lost)

When position is lost the time is the only parameter in the message that continues to update; all other parameters remain the same.

Command Format Enable/disable continuous time updating

To enable continuous time updating:

```
$JMODE,TIMEKEEP,YES<CR><LF>
```

To disable continuous time updating:

```
$JMODE,TIMEKEEP,NO<CR><LF>
```

Query the current setting

```
$JMODE,TIMEKEEP<CR><LF>
```

Receiver Response Response to issuing command to enable/disable continuous time updating

```
$>
```

Response to querying the current setting

If setting is currently enabled the response is:

```
$>JMODE,TIMEKEEP,ON
```

If setting is currently disabled the response is:

```
$>JMODE,TIMEKEEP,OFF
```

Additional Information

JMODE,TUNNEL Command

Command Type [General Operation and Configuration](#)

Description	Enable/disable faster reacquisition after coming out of a tunnel or query the current setting
--------------------	---

Command Format	Enable/disable faster reacquisition after coming out of a tunnel
-----------------------	--

To enable faster reacquisition:

```
$JMODE , TUNNEL , YES<CR><LF>
```

To disable faster reacquisition:

```
$JMODE , TUNNEL , NO<CR><LF>
```

Query the current setting

```
$JMODE , TUNNEL<CR><LF>
```

Receiver Response	Response to issuing command to turn functionality on/off
--------------------------	--

```
$>
```

Response to querying the current setting
--

If setting is currently enabled the response is:

```
$>JMODE , TUNNEL , ON
```

If setting is currently disabled the response is:

```
$>JMODE , TUNNEL , OFF
```

Additional Information

JMSG99 Command

Type [Crescent Vector](#)

Description Change the output in the [Bin99](#) message to be from the specified antenna

Format \$JMSG99 , 0

 where '0' is used view the primary antenna SNR (default)

 \$JMSG99 , 1

 where '1' is used view the secondary antenna SNR

**Receiver
Response** \$>

Other

JNMEA,GGAALLGNSS Command

Command Type [GLONASS](#)

Description Configure the GGA string to include full GNSS information (the number of used GLONASS satellites will be included in the [GPGLGA](#) message) or query the current setting

The GGA message is only supposed to report position and satellite information based on the GPS constellation. The combined GPS and GLONASS position and satellite data should be reported in the GNSS message, but some users with older equipment cannot utilize this message. This command allows users with older equipment that require a GGA message to be able to utilize and take advantage of the larger constellation of GPS and GLONASS satellites.

Command Format Include/exclude full GNSS information in GGA string

To include full GNSS information in GGA string:

```
$JNMEA,GGAALLGNSS,YES<CR><LF>
```

To exclude full GNSS information from GGA string:

```
$JNMEA,GGAALLGNSS,NO<CR><LF>
```

Query the current setting

```
$JNMEA,GGAALLGNSS<CR><LF>
```

Receiver Response Include/exclude full GNSS information in GGA string

```
$>
```

Query the current setting

If set to yes, querying the current setting returns the following:

```
$>JNMEA,GGAALLGNSS,YES
```

If set to no, querying the current setting returns the following:

```
$>JNMEA,GGAALLGNSS,NO
```

Additional Information

JNMEA,PRECISION Command

Command Type [GPS](#), [Local Differential and RTK](#), [OmniSTAR](#)

Description Specify or query the number of decimal places to output in the [GPGGA](#) and the [GPGLL](#) messages or query the current setting

Command Format Specify the number of decimal places

`$JNMEA,PRECISION,x<CR><LF>`

where 'x' specifies the number of decimal places from 1 to 8

Query the current setting

`$JNMEA,PRECISION<CR><LF>`

Receiver Response Specify the precision

`$>`

Query the current setting

`$>JNMEA,PRECISION,x`

where 'x' refers to the number of decimal places to output

Additional Information When using RTK or OmniSTAR HP/XP, Hemisphere GPS recommends you set JNMEA,PRECISION to at least 7 decimal places. High accuracy positioning techniques require at least 7 decimal places to maintain millimeter (mm) accuracy.

This command is the same as [JNP](#).

JNP Command

Command Type [GPS](#), [Local Differential and RTK](#), [OmniSTAR](#)

Description Specify or query the number of decimal places to output in the [GPGGA](#) and the [GPGLL](#) messages or query the current setting

Command Format Specify the number of decimal places

\$JNP , x <CR> <LF>

where 'x' specifies the number of decimal places from 1 to 8

Query the current setting

\$JNP <CR> <LF>

Receiver Response Specify the number of decimal places to output

\$>

Query the current setting

\$>JNP , x

where 'x' refers to the number of decimal places to output

Additional Information When using RTK or OmniSTAR HP/XP, Hemisphere GPS recommends you set JNP to at least 7 decimal places. High accuracy positioning techniques require at least 7 decimal places to maintain millimeter (mm) accuracy.

This command is the same as [JNMEA,PRECISION](#).

JOFF Command

Command Type [GPS](#)

Description Turn off all data messages being output through the current port or other port (or Port C), including any binary messages such as [Bin95](#) and [Bin96](#)

Command Format \$JOFF[,OTHER]<CR><LF>

When you specify the ',OTHER' data field (without the brackets), this command turns off all messages on the other port. There are no variable data fields for this message.

You can issue this command as follows to turn off all messages on Port C:

\$JOFF,PORTC<CR><LF>

Receiver Response \$>

Additional Information To turn off all data messages being output through all ports, including any binary messages such as Bin95 and Bin96, see the [JOFF,ALL](#) command

JOFF,ALL Command

Command Type [GPS](#)

Description Turn off all data messages being output through [all ports](#), including any binary messages such as [Bin95](#) and [Bin96](#)

Command Format \$JOFF,ALL<CR><LF>

Receiver Response \$>

Additional Information To turn off all data messages being output through a single port, including any binary messages such as Bin95 and Bin96, see the [JOFF](#) command

JOMS Command

Command Type [OmniSTAR](#)

Description Request the raw OmniSTAR subscription information

Command Format \$JOMS

Receiver Response \$>JOMS,Opt,Source,Type,AccrReduction,SDate,EDate,HourGlass,ExtTime,LinkVector,SW

Response Component	Description
Opt	VBS subscription: Indicates a WET or DRY subscription HP/XP subscription: Subscribed service level (varies according to the services subscribed from OmniSTAR, contents defined by OmniSTAR)
Source	VBS subscription: RTCM source ID, VBS, or VRC HP/XP subscription: Not applicable
Type	Subscription type (VBS, XP, HP, G2 etc; contents defined by OmniSTAR)
AccrReduction	<i>Not used</i>
SDate	Subscription start date
EDate	Subscription end date
HourGlass	Seconds of metered time
ExtTime	Seconds of extension
LinkVector	Hexadecimal mask of links
SW	OmniSTAR library version, contents defined by OmniSTAR

Example "Opt = VBS subscription" example

```
$>JOMS,DRY,ALL,VBS,0,01/06/2000,01/06/2001,0,0,1E00,1.43
```

"Opt = HP/XP subscription" example

```
$>JOMS,HPG2-GLO,0,GPG2,0,09/01/2010,09/01/2010,0,0,FFFFFFFF,HP 5.22
```

Additional Information

JPOS Command

Command Type [General Operation and Configuration](#)

Description Speed up the initial acquisition when changing continents with the receiver or query the receiver for the current position of the receiver (for example, powering up the receiver for the first time in Europe after it has been tested in Canada)

The command enables the receiver to begin the acquisition process for the closest SBAS spot beams. This saves some time with acquisition of the SBAS service. However, use of this message is typically not required because of the quick overall startup time of the receiver module.

Command Format Specify the latitude and longitude

\$JPOS , LAT , LON <CR> <LF>

where both 'LAT' and 'LON':

- Must be entered in decimal degrees
- Do not need to be more accurate than half a degree

Query the current setting

\$JPOS <CR> <LF>

Receiver Response Receiver response when specifying the latitude and longitude

\$>

Receiver response when querying the current setting

\$>JPOS , LAT , LON

Additional Information The status of this command is also output in the [JSHOW](#) message.

JQUERY,GUIDE Command

Command Type [General Operation and Configuration](#)

Description Query the receiver for its determination on whether or not it is providing suitable accuracy after both the SBAS and GPS have been acquired (up to five minutes)

This feature takes into consideration the download status of the SBAS ionospheric map and also the carrier phase smoothing of the unit.

Command Format \$JQUERY, GUIDE<CR><LF>

Receiver Response If the receiver is ready for use with navigation, or positioning with optimum performance, it returns:

\$>JQUERY, GUIDE, YES<CR><LF>

Otherwise, it returns:

\$>JQUERY, GUIDE, NO<CR><LF>

Additional Information

JQUERY,RTKSTAT Command

Command Type [Local Differential and RTK](#)

Description Perform a one-time query of the most relevant parameters affecting RTK

Command Format \$JQUERY,RTKSTAT<CR><LF>

As an alternative you can log this as a message using the [JASC,PSAT,RTKSTAT](#) command.

Receiver Response \$>JQUERY,RTKSTAT,MODE,TYP,AGE,SUBOPT,DIST,SYS,NUM,SNR,RSF,BSF,HAG*CC

where

Message Component	Description
MODE	FIX,FLT,DIF,AUT,NO
TYP	DFX,ROX,CMR,RTCM3,CMR+,...
AGE	Age of differential corrections, in seconds
SUBOPT	Subscription code (see Interpreting the \$JI and \$JK 'Date'/Subscription Codes to determine the meaning of the subscription code)
DIST	Distance to base in kilometers
SYS	Systems in use: <ul style="list-style-type: none">• GPS: L1, L2, L5• GLONASS: G1, G2• Galileo: E5a, E5b, E5a+b, E6
NUM	Number of satellites used by each system
SNR	Quality of each SNR path, where: <ul style="list-style-type: none">• A is > 20 dB• B is > 18 dB• C is > 15 dB• D is <= 15 dB
RSF	Rover slip flag (non zero if parity errors in last 5 minutes, good for detecting jamming and TCXO issues)
BSF	Base slip flag
HAG	Horizontal accuracy guess
*CC	Checksum

<CR>	Carriage return
<LF>	Line feed

Example \$>JQUERY,RTKSTAT,FIX,ROX,1,007F,0.0,(,L1,L2,G1,G2,),(,11,9,6,6,),(,A,A,A,B,),0,0
 ,0.009,000

**Additional
Information**

JRAIM Command

Command Type [RAIM](#)

Description Specify the parameters of the RAIM scheme that affect the output of the [PSAT.GBS](#) message or query the current setting

Command Format Specify the parameters of the RAIM scheme

```
$JRAIM,HPR,probHPR,probFALSE<CR><LF>
```

where:

Command Component	Description
HPR	Horizontal Protection Radius: notification in the PSAT.GBS message that the horizontal error has exceeded this amount will be received. The acceptable range for this value is 1 to 10,000 m. The default is 10 m.
probHPR	Maximum allowed probability that the position computed lies outside the HPR. The acceptable range for this value is 0.001% to 50%. The default is 5%.
probFALSE	Maximum allowed probability that there is a false alarm (that the position error is reported outside the of the HPR, but it is really within the HPR). The acceptable range for this value is 0.001% to 50%. The default is 1%.

Query the current setting

```
$JRAIM
```

Receiver Response Response to issuing command to specify RAIM scheme parameters

```
$>
```

Response to querying the current setting

```
$>JRAIM,HPR,probHPR,probFALSE
```

Example To specify the RAIM scheme parameters as HPR = 8 m, probHPR = 2%, and probFALSE = 0.5% issue the following command:

```
$JRAIM,8,2,0.5<CR><LF>
```

If you then query the receiver for the RAIM scheme issue the following command:

```
$JRAIM<CR><LF>
```

...and the response will be:

```
$>JRAIM,8.00,2.0000,0.5000
```

Additional Information The purpose of the probability of false alarm is to help make a decision on whether to declare a fault or warning in an uncertain situation. The philosophy is to only issue a fault if the user is certain (to within the probability of a false alarm) that the protection radius has been exceeded, else issue a warning.

JRAD Command Overview

This topic provides information related to the NMEA 0183 messages accepted by the receiver's e-Dif application. The following table provides a brief description of the commands supported by the e-Dif application for its control and operation.

Command	Description
JRAD,1	Display the current reference position in e-Dif applications only
JRAD,1,LAT,LON,HEIGHT	Use this command—a derivative of the JRAD,1,P command—when absolute positioning is required in e-Dif applications only
JRAD,1,P	<p>e-Dif: Record the current position as the reference with which to compute e-Dif corrections. This would be used in relative mode as no absolute point information is specified.</p> <p>DGPS Base Station: Record the current position as the reference with which to compute Base Station corrections in e-Dif applications only. This would be used in relative mode as no absolute point information is specified</p>
JRAD,2	Forces the receiver to use the new reference point (you normally use this command following a JRAD,1 type command)
JRAD,3	Invoke the e-Dif function once the unit has started up with the e-Dif application active, or, update the e-Dif solution (calibration) using the current position as opposed to the reference position used by the JRAD,2 command
JRAD,7	Turn auto recalibration on or off
JRAD,9.1.1	Initialize the Base Station feature and use the previously entered point, either with \$JRAD,1,P or \$JRAD,1,LAT,LON,HEIGHT , as the reference with which to compute Base Station corrections in e-Dif applications only. Use this for both relative mode and absolute mode.

Note: Use the [JSAVE](#) command to save changes you need to keep and wait for the \$>SAVE COMPLETE response.

JRAD,1 Command

Command Type [e-Dif, DGPS Base Station](#)

Description Display the current reference position in e-Dif applications only

Command Format \$JRAD,1<CR><LF>

Receiver Response \$>JRAD,1,LAT,LON,HEIGHT

where:

Command Component	Description
LAT	Latitude of the reference point in decimal degrees
LON	Longitude of the reference point in decimal degrees
HEIGHT	Ellipsoidal height of the reference point in meters

Upon startup of the receiver with the e-Dif application running—as opposed to with the SBAS application—no reference position will be present in memory. If you attempt to query for the reference position, the receiver's response will be:

\$>JRAD,1,FAILED,PRESENT LOCATION NOT STABLE

Example When you issue the \$JRAD,1 command the response will be similar to the following:

\$>JRAD,1,51.00233513,-114.08232345,1050.212

Additional Information

JRAD,1,LAT,LON,HEIGHT Command

Command Type [e-Dif](#), [DGPS Base Station](#)

Description Use this command—a derivative of the [JRAD,1,P](#) command—when absolute positioning is required in e-Dif applications only

Command Format \$JRAD , 1 , LAT , LON , HEIGHT<CR><LF>

where:

Command Component	Description
LAT	Latitude of the reference point in decimal degrees
LON	Longitude of the reference point in decimal degrees
HEIGHT	<p>Ellipsoidal height of the reference point in meters. Ellipsoidal height can be calculated by adding the altitude and the geoidal separation, both available from the GPGGA message.</p> <p>Example:</p> <p>\$GPGGA,173309.00,5101.04028,N,11402.38289,W,2,07,1.4,1071.0,M,-17.8,M,6.0,0122*48</p> <p>ellipsoidal height = 1071.0 + (-17.8) = 1053.2 meters</p>

Both latitude and longitude must be entered as decimal degrees. The receiver will not accept the command if there are no decimal places.

Receiver Response \$>JRAD , LAT , LON , HEIGHT

Additional Information

JRAD,1,P Command

Command Type [e-Dif](#), [DGPS Base Station](#)

Description **e-Dif:** Record the current position as the reference with which to compute e-Dif corrections. This would be used in relative mode as no absolute point information is specified.

DGPS Base Station: Record the current position as the reference with which to compute Base Station corrections in e-Dif applications only. This would be used in relative mode as no absolute point information is specified

Command Format \$JRAD,1,P<CR><LF>

Receiver Response \$>JRAD,1,OK

Additional Information

JRAD,2 Command

Command Type [e-Dif](#)

Description Forces the receiver to use the new reference point
You normally use this command following a [JRAD,1](#) type command.

Command Format \$JRAD , 2<CR><LF>

Receiver Response \$>JRAD , 2 , OK

Additional Information

JRAD,3 Command

Command Type [e-Dif](#)

Description This command has two primary purposes.

- To invoke the e-Dif function once the unit has started up with the e-Dif application active
- To update the e-Dif solution (calibration) using the current position as opposed to the reference position used by the [JRAD,2](#) command

Command Format \$JRAD , 3 <CR> <LF>

Receiver Response If the receiver has tracked enough satellites for a long enough period before you issue this command, it will respond with the following. (The tracking period can be from 3 to 10 minutes and is used for modeling errors going forward.

\$>JRAD , 3 , OK <CR> <LF>

If the e-Dif algorithms do not find sufficient data, the receiver responds with:

\$>JRAD , 3 , FAILED , NOT ENOUGH STABLE SATELLITE TRACKS

Additional Information If you receive the failure message after a few minutes of operation, try again shortly after until you receive the "OK" acknowledgement message. The e-Dif application begins operating as soon as the \$>JRAD,3,OK message has been received; however, a you will still need to define a reference position for e-Dif unless relative positioning is sufficient for any needs.

JRAD,7 Command

Command Type [e-Dif](#)

Description Turn auto recalibration on or off

Command Format \$JRAD,7,n
where n = auto recalibration variable (0 = Off or 1 = On, 0 is the default)

Receiver Response \$>

Additional Information

JRAD,9,1,1 Command

Command Type [DGPS Base Station](#)

Description Initialize the Base Station feature and use the previously entered point, either with [\\$JRAD,1,P](#) or [\\$JRAD,1,LAT,LON,HEIGHT](#), as the reference with which to compute Base Station corrections in e-Dif applications only. Use this for both relative mode and absolute mode.

Command Format \$JRAD,9,1,1<CR><LF>

Receiver Response \$>JRAD,9,OK

Additional Information The [\\$JASC,RTCM,1](#) command must be sent to the receiver to start outputting standard RTCM corrections.

JRELAY Command

Command Type [General Operation and Configuration](#)

Description Send user-defined text out of a serial port

Command Format `$JRELAY , PORTx , MSG<CR><LF>`

- 'x' = destination port where the message (MSG) will be sent
- 'MSG' = message to be sent

Receiver Response `$>`

Example Command
`$JRELAY , PORTA , HELLO\nTHERE\n<CR><LF>`

Response
HELLO
THERE
\$>

Additional Information

JRESET Command

Command Type [General Operation and Configuration](#)

Description Reset the receiver to its default operating parameters by:

- Turning off outputs on all ports
- Saving the configuration
- Setting the configuration to its defaults (in following table)

Configuration	Setting
Elev Mask	5
Residual limit	10
Alt aiding	None
Age of Diff	45 minutes
Air mode	Auto
Diff type	Default for app
NMEA precision	5 decimals
COG smoothing	None
speed smoothing	None
WAAS	UERE thresholds

Command Format \$JRESET[,X]<CR><LF>

where 'X' is an optional field:

- When set to ALL does everything \$JRESET does, plus it clears almanacs
- When set to BOOT does everything \$JRESET,ALL does, plus clears use of the real-time clock at startup, clears use of backed-up ephemeris and almanacs, and reboots the receiver when done

Receiver Response \$JRESET
 \$> Saving Configuration. Please Wait...
 \$>
 \$> Save Complete

Additional Information **CAUTION:** \$JRESET clears all parameters. For the V101 Series and the LV101 you will have to issue the [\\$JATT, FLIPBRD, YES](#) command to properly redefine the circuitry orientation inside the product once the receiver has reset. Failure to do so will cause radical heading behavior.

JRTK Command Overview

The JRTK commands are used to define or query RTK settings.

Command	Description
JRTK,1	Show the receiver's reference position (can issue command to base station or rover)
JRTK,1,LAT,LON,HEIGHT	Set the receiver's reference position to the coordinates you enter (can issue command to base station or rover)
JRTK,1,P	Set the receiver's reference coordinates to the current calculated position if you do not have known coordinates for your antenna location (can issue command to base station or rover)
JRTK,5	Show the base station's transmission status for RTK applications (can issue command to base station)
JRTK,5,Transmit	Suspend or resume the transmission of RTK (can issue command to base station)
JRTK,6	Display the progress of the base station (can issue command to base station)
JRTK,12	Disable or enable the receiver to go into fixed integer mode (RTK) vs. float mode (L-Dif) - can issue command to rover
JRTK,17	Display the transmitted latitude, longitude, and height of the base station (can issue command to base station or rover)
JRTK,18	Display the distance from the rover to the base station, in meters (can issue command to rover)
JRTK,28	Set the base station ID transmitted in ROX/DFX/CMR/RTCM3 messages (can issue command to base station)

JRTK,1 Command

Command Type [Local Differential and RTK](#)

Description Show the receiver's reference position (can issue command to base station or rover)

Command Format \$JRTK,1<CR><LF>

Receiver Response \$JRTK,1,LAT,LON,HEIGHT
where

Command Component	Description
LAT	Latitude of the reference point in decimal degrees
LON	Longitude of the reference point in decimal degrees
HEIGHT	<p>You must enter HEIGHT as ellipsoidal height in meters. Ellipsoidal height can be calculated by adding the altitude and the geoidal separation, both available from the GPGGA message.</p> <p>Example:</p> <p>\$GPGGA,173309.00,5101.04028,N,11402.38289,W,2,07,1.4,1071.0,M,-17.8,M,6.0,0122*48</p> <p>ellipsoidal height = 1071.0 + (-17.8) = 1053.2 meters</p>

Example \$>JRTK,1,33.55679117,-111.88955483,374.600

Additional Information

JRTK,1,LAT,LON,HEIGHT Command

Command Type [Local Differential and RTK](#)

Description Set the receiver's reference position to the coordinates you enter (can issue command to base station or rover)

Command Format \$JRTK,1,LAT,LON,HEIGHT<CR><LF>

where:

Comm and Component	Description
LAT	Latitude of the reference point in decimal degrees
LON	Longitude of the reference point in decimal degrees
HEIGHT	<p>You must enter HEIGHT as ellipsoidal height in meters. Ellipsoidal height can be calculated by adding the altitude and the geoidal separation, both available from the GPGGA message.</p> <p>Example:</p> <p>\$GPGGA,173309.00,5101.04028,N,11402.38289,W,2,07,1.4,1071.0,M,-17.8,M,6.0,0122*48</p> <p>ellipsoidal height = 1071.0 + (-17.8) = 1053.2 meters</p>

Note: You must enter both latitude and longitude in decimal degrees; the receiver will not accept the command if there are no decimal places.

Receiver Response \$>

Additional Information

JRTK,1,P Command

Command Type [Local Differential and RTK](#)

Description Set the receiver's reference coordinates to the current calculated position if you do not have known coordinates for your antenna location (can issue command to base station or rover)

Command Format \$JRTK,1,P<CR><LF>

Receiver Response \$>

Additional Information If you have known coordinates for your antenna location, use the [JRTK,1,LAT,LON,HEIGHT](#) command to enter the latitude and longitude (in decimal degrees) and the ellipsoidal height (in meters).

JRTK,5 Command

Command Type [Local Differential and RTK](#)

Description Show the base station's transmission status for RTK applications (can issue command to base station)

Command Format \$JRTK,5<CR><LF>

Receiver Response If transmission status is suspended, response is as follows:
\$>JRTK,6

If transmission status is not suspended, response is as follows:
\$>JRTK,5,1

Additional Information Also see the [JRTK,6](#) command.

JRTK,5,Transmit Command

Command Type [Local Differential and RTK](#)

Description	Suspend or resume the transmission of RTK (can issue command to base station)
--------------------	---

Command Format	<code>\$JRTK,5,Transmit<CR><LF></code> where "Transmit" is 0 (suspend) or 1 (resume)
-----------------------	---

Receiver Response	If the transmission status is not suspended and you issue the following command to suspend:
--------------------------	---

`$JRTK,5,0<CR><LF>`

the response is as follows:

`$>JRTK,5,OK`

Similarly, if the transmission status is suspended and you issue the following command to resume:

`$JRTK,5,1<CR><LF>`

the response is again as follows:

`$>JRTK,5,OK`

Additional Information

JRTK,6 Command

Command Type [Local Differential and RTK](#)

Description Display the progress of the base station (can issue command to base station)

Command Format \$JRTK,6<CR><LF>

Receiver Response \$JRTK,6,TimeToGo,ReadyTransmit,Transmitting
where

Response Component	Description
TimeToGo	Seconds left until ready to transmit RTK
ReadyTransmit	Non zero when configured to transmit and ready to transmit RTK on at least one port. It is a bit mask of the transmitting port, with bit 0 being port A, bit 1 being port B and bit 2 being port C. It will be equal to "Transmitting" unless transmission has been suspended with \$JRTK,5,0.
Transmitting	Non-zero when actually transmitting RTK on at least one port. It is a bit mask of the transmitting port, with bit 0 being port A, bit 1 being port B and bit 2 being port C.

Example If the receiver is not ready to transmit:

\$>JRTK,6,263,0,0

If the receiver is currently transmitting on Port B:

\$>JRTK,6,0,2,2

Additional Information

JRTK,12 Command

Warning! Hemisphere GPS recommends that only advanced users employ this command.

Command Type [Local Differential and RTK](#)

Description Disable or enable the receiver to go into fixed integer mode (RTK) vs. float mode (L-Dif) - can issue command to rover

Note: Requires RTK rover subscription

Command Format \$JRTK,12,x

where x is:

- 1 = Allow RTK (recommended, and the default)
- 0 = Do not allow RTK, stay in L-Dif

Receiver Response \$>

Additional Information In high multipath conditions it may be desirable to prevent the rover from obtaining a fixed position. Using \$JRTK,12,0 while logging position data is useful for determining the level of multipath present.

JRTK,17 Command

Command Type [Local Differential and RTK](#)

Description Display the transmitted latitude, longitude, and height of the base station (can issue command to base station or rover)

Command Format \$JRTK,17<CR><LF>

Receiver Response \$>JRTK,17,lat,lon,height

Example \$>JRTK,17,33.55709242,-111.88916894,380.534

Additional Information Format is similar to [JRTK,1,LAT,LON,HEIGHT](#)

JRTK,18 Command

Command Type [Local Differential and RTK](#)

Description	Display the distance from the rover to the base station, in meters (can issue command to rover)
--------------------	---

Command Format	\$JRTK,18<CR><LF>
-----------------------	-------------------

Receiver Response	\$>JRTK,18,d where 'd' is the baseline distance in meters
--------------------------	--

Additional Information	
-------------------------------	--

JRTK,28 Command

Command Type [Local Differential and RTK](#)

Description Set the base station ID transmitted in ROX/DFX/CMR/RTCM3 messages (can issue command to base station), where:

- Default is 333
- Range is 0-4095 (except for CMR which is 0-31)

Command Format Set the base station ID

```
$JRTK,28,BASEID<CR><LF>
```

where 'BASEID' is the base station ID

Query the current setting

```
$JRTK,28<CR><LF>
```

Receiver Response \$>

Example To set the base station ID to 123 issue the following command:

```
$JRTK,28,123<CR><LF>
```

If the base station ID is 333 and you issue the \$JRTK,28<CR><LF> query the response is:

```
$>JRTK,28,333
```

Additional Information

JSAVE Command

Command Type [General Operation and Configuration](#)

Description Send this command after making changes to the operating mode of the receiver

Command Format \$JSAVE<CR><LF>

Receiver Response \$> SAVING CONFIGURATION. PLEASE WAIT...
then
\$> Save Complete

Additional Information Ensure that the receiver indicates that the save process is complete before turning the receiver off or changing the configuration further.

No data fields are required. The receiver indicates that the configuration is being saved and indicates when the save is complete.

JSHOW Command

Command Type [General Operation and Configuration](#)

Description Query the current operating configuration of the receiver

Command Format \$JSHOW[, SUBSET] <CR><LF>

Receiver Response Use the JSHOW command without the optional 'SUBSET' field to provide a complete response from the receiver.

Example:

```
$>JSHOW,BAUD,9600 (1)
$>JSHOW,BAUD,9600,OTHER (2)
$>JSHOW,BAUD,9600,PORTC (3)
$>JSHOW,ASC,GPGGA,1.0,OTHER (4)
$>JSHOW,ASC,GPVTG,1.0,OTHER (5)
$>JSHOW,ASC,GPGSV,1.0,OTHER (6)
$>JSHOW,ASC,GPGST,1.0,OTHER (7)
$>JSHOW,ASC,D1,1,OTHER (8)
$>JSHOW,DIFF,WAAS (9)
$>JSHOW,ALT,NEVER (10)
$>JSHOW,LIMIT,10.0 (11)
$>JSHOW,MASK,5 (12)
$>JSHOW,POS,51.0,-114.0 (13)
$>JSHOW,AIR,AUTO,OFF (14)
$>JSHOW,FREQ,1575.4200,250 (15)
$>JSHOW,AGE,1800 (16)
```

Description of responses:

Line	Description
1	Current port is set to a baud rate of 9600
2	Other port is set to a baud rate of 9600
3	Port C is set to a baud rate of 9600 (Port C is not usually connected externally on the finished product)
4	GPGGA is output at a rate of 1 Hz from the other port
5	GPVTG is output at a rate of 1 Hz from the other port
6	GPGSV is output at a rate of 1 Hz from the other port
7	GPGST is output at a rate of 1 Hz from the other port
8	D1 is output at a rate of 1 Hz from the other
9	Current differential mode is WAAS
10	Status of the altitude aiding feature (see the JALT command for information how to set turn altitude aiding on or off)
11	Receiver does not support this feature

12	Elevation mask cutoff angle (in degrees)
13	Current send position used for startup, in decimal degrees
14	Current status of the AIR mode (see the JAIR command for information how to set the AIR mode)
15	Current frequency of the L-band receiver 'AUTO' appears at the end of the query response only when the OmniSTAR receiver is in 'auto-tune' mode.
16	Current maximum acceptable differential age, in seconds (see the JAGE command for information how to set the differential age)

When you issue this command with the optional ',SUBSET' data field (without the brackets), a one-line response is provided. The subset field may be:

- CONF
- GP
- PORT

When you specify CONF for ',SUBSET' (without the brackets), an example response is:

```
$>JSHOW,CONF,N,0.0,10.0,5,A,60W
```

The following table explains the example response:

Message Component	Description
\$JSHOW,CONF	Message header
N	Indicates no altitude aiding
0.0	Indicates the aiding value, if specified (either height or PDOP threshold)
10.0	Residual limit for the \$JLIMIT command
5	Elevation mask cutoff angle (in degrees)
A	AIR mode indication
60	Maximum acceptable differential age (in seconds)
W	Current differential mode, 'W' indicates WAAS mode

When you specify GP for ',SUBSET,' (without the brackets) an example response is:

```
$>JSHOW,GP,GP,1.0
```

This response will provide the JSHOW,GP message header followed by each message currently being output through the current port and the update rate for that message.

When you specify PORT for ',SUBSET,' (without the brackets) and example response is:

```
$>JSHOW,THISPORT,PORTA
```

This response shows the port to which you are currently connected.

Example See "Receiver Response" section above

**Additional
Information**

JSMOOTH Command

Command Type [GPS](#)

Description Set the carrier smoothing interval (15 to 6000 seconds) or query the current setting
This command provides the flexibility to tune in different environments. The default for this command is 900 seconds (15 minutes) or LONG. A slight improvement in positioning performance (depending on the multipath environment) may occur if you use either the SHORT (300 seconds) or LONG (900 seconds) smoothing interval.

Command Format Set the carrier smoothing interval

To set the carrier smoothing interval to a specific number of seconds issue the following command:

```
$JSMOOTH,x<CR><LF>
```

where 'x' is one of the following:

- Number of seconds
- SHORT (equals 300 seconds)
- LONG (equals 900 seconds)

Query the current setting

```
$JSMOOTH<CR><LF>
```

It will return the word SHORT or LONG as well as the number of seconds used where:

- SHORT precedes the number of seconds for any setting less than 900 seconds
- LONG precedes the number of seconds for any setting greater than or equal to 900 seconds

Receiver Response Receiver response when setting the carrier smoothing interval

```
$>
```

Receiver response when querying the current carrier smoothing interval

```
$>JSMOOTH,x
```

Example To set the carrier smoothing interval to 750 seconds issue the following command:

```
$JSMOOTH,750<CR><LF>
```

...and if you then query the receiver using \$JSMOOTH the response will be:

```
$JSMOOTH,SHORT750
```

To set the carrier smoothing interval to 300 seconds (5 minutes) issue the following command:

```
$JSMOOTH,SHORT<CR><LF>
```

To set the carrier smoothing interval to 900 seconds (15 minutes) issue the following command:

```
$JSMOOTH,LONG<CR><LF>
```

Additional Information If you are unsure of the best value for this setting, leave it at the default setting of LONG (900 seconds). The status of this command is also output in the [JSHOW](#) message.

JT Command

Command Type [General Operation and Configuration](#)

Description Query the receiver for its GPS engine type

Command Format \$JT<CR><LF>

Receiver Response \$>JT,xxxx

where xxxx indicates the GPS engine and mode:

JT Command Response (xxxx)	GPS Engine	Mode
DF2b	Eclipse	WAAS, RTK Base
DF2g	Eclipse	OmniSTAR
DF2r	Eclipse	RTK Rover
DF3g	Eclipse II	WAAS, RTK Base
DF3i	Eclipse II	e-Dif
DF3r	Eclipse II	RTK Rover
SX2a	Crescent Vector	WAAS RTK
SX2b	Crescent	Base
SX2g	Crescent	WAAS
SX2i	Crescent	e-Dif
SX2r	Crescent	Rover

Example When you issue the \$JT<CR><LF> command a typical response may be:

\$>JT,DF2b,MX31rev=28

DF2b indicates an Eclipse receiver with WAAS and RTK Base functionality.

Note: MX31rev=28 is the processor type.

Additional Information

JTAU Command Overview

The JTAU command is used to set the time constants for specific parameters for Crescent, Crescent Vector, and Eclipse products.

Command	Description
JTAU,COG	Set the course over ground time (COG) constant and query the current setting
JTAU,SPEED	Set the speed time constant and query the current setting

JTAU,COG Command

Note: The [JATT,COGTAU](#) command provides identical functionality but works only with Crescent Vector products.

Command Type [GPS](#)

Description Set the course over ground (COG) time constant (0.00 to 3600.00 seconds) or query the current setting

This command allows you to adjust the level of responsiveness of the COG measurement provided in the [GPVTG](#) message. The default value is 0.00 seconds of smoothing. Increasing the COG time constant increases the level of COG smoothing.

Command Format Set the COG time constant

\$JTAU,COG,tau<CR><LF>

where 'tau' is the new COG time constant that falls within the range of 0.00 to 200.00 seconds

The setting of this value depends upon the expected dynamics of the Crescent. If the Crescent will be in a highly dynamic environment, this value should be set lower because the filtering window would be shorter, resulting in a more responsive measurement. However, if the receiver will be in a largely static environment, this value can be increased to reduce measurement noise.

Query the current setting

\$JTAU,COG<CR><LF>

Receiver Response Receiver response when setting the COG time constant

\$>

Receiver response when querying the current COG time constant

\$>JTAU,COG,tau<CR><LF>

Example To set the COG time constant as 2 seconds issue the following command:

\$JTAU,COG,2<CR><LF>

Additional Information You can use the following formula to determine the COG time constant:

$\text{tau (in seconds)} = 10 / \text{maximum rate of change of course (in } ^\circ/\text{s)}$

If you are unsure about the best value for this setting, it is best to be conservative and leave it at the default setting of 0.00 seconds.

JTAU,SPEED Command

Note: The [JATT,SPDTAU](#) command provides identical functionality but works only with Crescent Vector products.

Command Type [GPS](#)

Description Set the speed time constant (0.00 to 3600.00 seconds) or query the current setting

This command allows you to adjust the level of responsiveness of the speed measurement provided in the [GPVTG](#) message. The default value is 0.00 seconds of smoothing. Increasing the speed time constant increases the level of speed measurement smoothing.

Command Format Set the speed time constant

```
$JTAU,SPEED,tau<CR><LF>
```

where 'tau' is the new speed time constant that falls within the range of 0.0 to 200.00 seconds

The setting of this value depends upon the expected dynamics of the receiver. If the receiver will be in a highly dynamic environment, you should set this to a lower value, since the filtering window will be shorter, resulting in a more responsive measurement. However, if the receiver will be in a largely static environment, you can increase this value to reduce measurement noise.

Query the current setting

```
$JTAU,SPEED<CR><LF>
```

Receiver Response Receiver response when setting the speed time constant

```
$>
```

Receiver response when querying the current speed time constants

```
$>JTAU,SPEED,tau<CR><LF>
```

Example To set the speed time constant as 4.6 seconds issue the following command:

```
$JTAU,SPEED,4.6<CR><LF>
```

Additional Information You can use the following formula to determine the COG time constant (Hemisphere GPS recommends testing how the revised value works in practice):

$\text{tau (in seconds)} = 10 / \text{maximum acceleration (in m/s}^2\text{)}$

If you are unsure of the best value for this setting, it is best to be conservative and leave it at the default setting:

- Crescent Vector receivers: default of 0.0 seconds
- Non-Crescent Vector receivers: default of LONG (900 seconds)

JWAASPRN Command

Command Type [SBAS](#)

Description Change the SBAS PRNs in memory or query the receiver for current PRNs in memory

Valid PRNs include:

- EGNOS: 120, 124, 126
- GAGAN: 127
- MSAS: 129, 137
- WAAS: 133, 135, 138

Command Format Change the SBAS PRNs in memory

`$JWAASPRN, PRN1, PRN2, PRN3<CR><LF>`

where 'PRN1' and 'PRN2' specify PRNs for Crescent receivers and 'PRN3' specifies the additional PRN for Eclipse receivers

Query the current setting

`$JWAASPRN<CR><LF>`

Receiver Response Response to issuing command to change PRNs

`$>`

Response to querying the current setting

`$>JWAASPRN, PRN1, PRN2 [, PRN3]`

Example To change the SBAS PRNs in memory for an Eclipse receiver to WAAS PRNs (133, 135, 138) issue the following command:

`$>JWAASPRN, 133, 135, 138<CR><LF>`

Additional Information You can specify an auto-tune mode to tune to the appropriate SBAS PRNs based on the autonomous GPS position. To auto-tune the PRNs issue the following command:

`$JWAASPRN, AUTO`

If you then query the receiver for the PRNs the receiver response will show 'AUTO' at the end. For example, if you query the receiver and the PRNs are 133,135, and 138 and autotuning is enabled the response is as follows:

`$>JWAASPRN, 133, 135, 138, AUTO`

PCSI,0 Command (Receiver Help Query command)

Command Type [NMEA 0183 SBX](#)

Description Hemisphere GPS proprietary NMEA 0183 query
Query the SBX to output a list of available proprietary PCSI commands

Command Format \$PCSI,0<CR><LF>

Receiver Response \$PCSI,ACK,0
\$PCSI,P003-OK,012
\$PCSI,0 ->HELP Msg
\$PCSI,1 ->Status line A,<T>,<S>
\$PCSI,2 ->Status line B,<T>
\$PCSI,3 ->Dump Search,<x>
\$PCSI,4 ->Wipe Search
\$PCSI,5 ->Port Rate,<P0>,<P1>
\$PCSI,6 ->Reset
\$PCSI,7 ->RTCM Mode

Additional Information

PCSI,1 Command (Status Line A, Channel 0 command)

Command Type [NMEA 0183 SBX](#)

Description Hemisphere GPS proprietary NMEA 0183 query
Query the SBX for a selection of parameters related to the operational status of its primary channel

Command Format \$PCSI,1<CR><LF>

Receiver Response \$PCSI,ACK,1
\$PCSI,CS0,PXXX-Y.YYY,SN,fff.f,M,ddd,R,SS,SNR,MTP,WER,ID,H,T,G

where:

Response Component	Description
CS0	Channel 0
PXXX-Y.YYY	Resident SBX firmware version
SN	SBX receiver serial number
fff.f	Channel 0 current frequency
M	Frequency mode (A = automatic, M = manual, D = database)
ddd	MSK bit rate
R	RTCM rate mode (A = automatic, M = manual, D = database)
SS	Signal strength
SNR	Signal-to-noise ratio
MTP	Message throughput
WER	Word Error Rate - Percentage of bad 30-bit RTCM words in the last 25 words
ID	Beacon ID to which the receiver's primary channel is tuned
H	Health of the tuned beacon [0-7]
T	\$PCSI,1 status output period [0-99]
G	AGC gain in dB (0 to 48 db)

Additional Information Optionally you can modify the Status Line A query to request the output of the response message once every period at a specified output rate. It has the following format, where 'T' is the output period in seconds:

\$PCSI,1,T<CR><LF>

The response will be:

```
$PCSI , ACK , 1  
$PCSI , CS0 , PXXXY . YYY , SN , f f f . f , M , d d d , R , SS , SNR , MTP , WER , ID , H , T , G
```

You can stop the output of the message by either of the following:

- Cycling receiver power
- Issuing the \$PCSI,1<CR><LF> query without the output period field

The response message has the same format as discussed above. In addition to this modified version of the Status Line A command, an additional 'S' field may be placed after the 'T' field, resulting in the following command:

```
$PCSI , 1 , T , S <CR> <LF>
```

The 'S' field is not a variable and specifies that the output of the Status Line A message should continue after the power has been cycled. To return the receiver to the default mode (in which message output ceases after receiver power is cycled) send the \$PCSI,1<CR><LF> query to the receiver.

You may send the \$PCSI,1 query through either serial port for reporting of the full status of the primary receiver channel. The query response is returned to the port from which you issued the command. When querying the primary receiver channel using the secondary serial port, no interruptions in RTCM data output will occur on the primary port provided the SBX has acquired a valid beacon.

The response is different depending on whether you are connected directly to the SBX-4 or not.

- If connected directly (by hardware or [JCONN](#)), the response will be both an acknowledgement as well as the full PCSI,1 message.
 - If connected through a Crescent receiver (such as the R110) you may see the full PCSI,1 message. Consider [PCSI,1.1](#) to generate periodic output.
-

PCSI,1,1 Command (Beacon Status command)

Command Type [Beacon Receiver](#)

Description Obtain PCSI,CS0 beacon status data from an SBX engine when interfaced to the receiver Port D. When you send this command through either Port A, B, or C it is automatically routed to Port D. The resulting PCSI,CS0 message is returned to the same port from which the command was sent at the desired rate.

Command Format \$PCSI,1,1<CR><LF>

Receiver Response \$PCSI,CS0,Pxxx-y.yyy,SN,fff.f,M,ddd,R,SS,SNR,MTP,WER,ID,H,T,G

Example:

\$PCSI,CS0,P030-0.000,19001,313.0,D,100,D,18,8,80,0,63,0,1,48

where:

Response Component	Description
CS0	Channel 0
PXXX-Y.YYY	Resident SBX firmware version
SN	SBX receiver serial number
fff.f	Channel 0 current frequency
M	Frequency mode (A = automatic, M = manual, D = database)
ddd	MSK bit rate
R	RTCM rate mode (A = automatic, M = manual, D = database)
SS	Signal strength
SNR	Signal-to-noise ratio
MTP	Message throughput
WER	Word Error Rate - Percentage of bad 30-bit RTCM words in the last 25 words
ID	Beacon ID to which the receiver's primary channel is tuned
H	Health of the tuned beacon (0-7)
T	\$PCSI,1 status output period (0-99)
G	AGC gain in, dB (0 to 48)

Additional Information

PCSI,2 Command (Status Line B, Channel 1 command)

Command Type [NMEA 0183 SBX](#)

Description Hemisphere GPS proprietary NMEA 0183 query
Query the SBX to output a selection of parameters related to the operational status of its secondary channel

Command Format \$PCSI , 2<CR><LF>

Receiver Response \$PCSI , ACK , 2
\$PCSI , CS1 , PXXX-Y.YYY , SN , fff.f , M , ddd , R , SS , SNR , MTP , WER , ID , H , T

where:

Response Component	Description
CS1	Channel 1
PXXX-Y.YYY	Resident SBX firmware version
SN	SBX receiver serial number
fff.f	Channel 1 current frequency
M	Frequency Mode (A = automatic, M = manual, D = database)
ddd	MSK bit rate
R	RTCM rate mode (A = automatic, M = manual, D = database)
SS	Signal strength
SNR	Signal to noise ratio
MTP	Message throughput
WER	Word Error Rate - Percentage of bad 30-bit RTCM words in the last 25 words
ID	Beacon ID to which the receiver's secondary channel is tuned
H	Health of the tuned beacon (0-7)
T	\$PCSI,1 status output period (0-99)

Additional Information Optionally you can modify the Status Line B query to request the output of the response message once every period. It has the following format, where T is the output period in seconds:

\$PCSI , 2 , T<CR><LF>

The response will:

\$PCSI , ACK , 2

\$PCSI,CS0,PXXX-Y.YYY,SN,fff.f,M,ddd,R,SS,SNR,MTP,WER,ID,H,T

The response message has the same format as discussed above. The Status Line B message output cannot be set to remain active after the power of the SBX has been cycled.

The \$PCSI,2 query may be sent through the either serial port for reporting of the full status of the secondary receiver channel. The response to the query is returned to the port from which the command was issued. When querying the secondary receiver channel using the secondary serial port, no interruptions in RTCM data output will occur on the primary port provided that SBX has acquired a valid beacon.

PCSI,3,1 Command (Receiver Search Dump command)

Command Type [NMEA 0183 SBX](#)

Description Hemisphere GPS proprietary NMEA 0183 query
Query the SBX to output the search information used for beacon selection in Automatic Beacon Search mode. The output has three frequencies per line.

Command Format \$PCSI,3,1<CR><LF>

Receiver Response \$PCSI,ACK,3,1
\$PCSI,tag1,freq1,ID1,chan1,snr1,ss1,tag2,freq2,ID2,chan2,snr2,ss2,tag3,freq3,ID3,chan3,snr3,ss3

where:

Response Component	Description
tag	Channel number with a range of 1 to 84
freq	Channel frequency (kHz * 10)
ID	Beacon ID
chan	Channel information
snr	SNR (dB)
ss	Signal Strength (dBuV/m)

Example

```
$PCSI,ACK,3,1
$PCSI,01,2835,209,0E,00,-0009,02,2840,339,0E,00,-0012,03,2845,006,0E,00,0009
$PCSI,04,2850,342,0E,00,-0010,05,2855,547,0E,00,-0005,06,2860,109,0E,00,-0011
$PCSI,07,2865,188,0E,00,-0007,08,2870,272,0E,00,-0004,09,2875,682,0E,00,-0006
$PCSI,10,2880,645,0E,00,-0007,11,2885,256,0E,00,-0009,12,2890,000,06,00,-0012
$PCSI,13,2895,132,0E,00,-0009,14,2900,281,0E,00,-0010,15,2905,634,0E,00,-0008
$PCSI,16,2910,172,0E,00,-0007,17,2915,006,0E,00,-0009,18,2920,546,0E,00,-0014
$PCSI,19,2925,358,0E,00,-0008,20,2930,479,0E,00,-0009,21,2935,358,0E,00,-0011
$PCSI,22,2940,853,0E,00,-0005,23,2945,588,0E,00,-0015,24,2950,210,0E,00,-0011
$PCSI,25,2955,000,06,00,-0011,26,2960,663,0E,00,-0010,27,2965,596,0E,00,-0009
$PCSI,28,2970,000,06,00,-0009,29,2975,917,0E,00,-0009,30,2980,000,06,00,-0016
$PCSI,31,2985,343,0E,00,-0013,32,2990,546,0E,00,-0010,33,2995,546,0E,00,-0010
$PCSI,34,3000,172,0E,00,-0014,35,3005,006,0E,00,-0011,36,3010,1006,0E,00,-0009
$PCSI,37,3015,006,0E,00,-0015,38,3020,300,0E,00,-0013,39,3025,277,0E,00,-0100
$PCSI,40,3030,479,0E,00,-0010,41,3035,006,0E,00,-0012,42,3040,050,0E,00,-0008
$PCSI,43,3045,000,06,00,-0014,44,3050,172,0E,00,-0013,45,3055,000,06,00,-0011
$PCSI,46,3060,000,06,00,-0011,47,3065,000,06,00,-0014,48,3070,000,06,00,-0010
$PCSI,49,3075,000,06,00,-0012,50,3080,006,0E,00,-0015,51,3085,000,06,00,-0015
$PCSI,52,3090,300,0E,00,-0007,53,3095,000,06,00,-0013,54,3100,000,06,00,-0013
$PCSI,55,3105,000,06,00,-0012,56,3110,127,0E,00,-0013,57,3115,000,06,00,-0012
$PCSI,58,3120,596,0E,00,-0012,59,3125,051,0E,00,-0009,60,3130,000,06,00,-0011
$PCSI,61,3135,213,0E,00,-0008,62,3140,000,06,00,-0011,63,3145,000,06,00,-0015
$PCSI,64,3150,302,0E,00,-0008,65,3155,000,06,00,-0009,66,3160,000,06,00,-0003
$PCSI,67,3165,000,06,00,-0013,68,3170,000,06,00,-0011,69,3175,612,0E,01,0000
$PCSI,70,3180,000,06,00,-0015,71,3185,000,06,00,-0008,72,3190,000,06,00,-0009
$PCSI,73,3195,000,06,00,0011,74,3200,1002,0E,01,-0002,75,3205,067,0E,00,-0008
$PCSI,76,3210,001,0E,00,-0008,77,3215,000,06,00,-0009,78,3220,132,0E,00,-0009
$PCSI,79,3225,000,06,00,-0010,80,3230,339,0E,00,-0013,81,3235,000,06,00,-0011
$PCSI,82,3240,000,06,00,-0010,83,3245,202,0E,00,-0007,84,3250,006,0E,00,-0002
```

Additional Information

PCSI,3,2 Command (Ten Closest Stations command)

Command Type [Beacon Receiver](#)

Description Display the ten closest beacon stations

Command Format \$PCSI,3,2<CR><LF>

Receiver Response \$PCSI,3,2,StationID,name,freq,status,distance,health,WER
 \$PCSI,3,2, ...
 \$PCSI,3,2, ...
 \$PCSI,3,2, ...
 \$PCSI,3,2, ...
 ...

where:

Response Component	Description
StationID	Specific ID number for beacon stations (appears in the last field of the GPGGA message)
name	Name of station and will display time/date of update for a station added by using information from an almanac message (in the format ddmmyy->time)
freq	Frequency, in kHz (scaled by 10), on which the station is transmitting. In the first line of the Example below, 2870 indicates 287.0 kHz.
status	0 (operational), 1 (undefined), 2 (no information), 3 (do not use)
distance	Calculated in nautical miles
health	-1 (not updated), 8 (undefined), 0-7 (valid range)
WER	-1 (not updated), 0-100 (valid range)

Example

```
$PCSI,3,2, 849,Polson      MT,2870,0,210,0,0,-1,-1
$PCSI,3,2, 848,Spokane    WA,3160,0,250,0,0,-1,-1
$PCSI,3,2, 907,Richmond    BC,3200,0,356,0,0,-1,-1
$PCSI,3,2, 888,Whidbey Is. WA,3020,0,363,0,0,-1,-1
$PCSI,3,2, 887,Robinson Pt. WA,3230,0,383,0,0,-1,-1
$PCSI,3,2, 874,Billings    MT,3130,0,389,0,0,-1,-1
$PCSI,3,2, 871,Appleton    WA,3000,0,420,0,0,-1,-1
$PCSI,3,2, 908,Amphitrite Pt BC,3150,0,448,0,0,-1,-1
$PCSI,3,2, 886,Fort Stevens OR,2870,0,473,0,0,-1,-1
$PCSI,3,2, 909,Alert Bay   BC,3090,0,480,0,0,-1,-1
```

Additional Information

PCSI,3,3 Command (Station Database command)

Command Type [Beacon Receiver](#)

Description Display the contents of the beacon station database

Command Format \$PCSI,3,3<CR><LF>

Receiver Response \$PCSI,3,3,IDref1,IDref2,StationID,name,freq,lat,long,datum,status
 \$PCSI,3,3, ...
 \$PCSI,3,3, ...
 \$PCSI,3,3, ...
 \$PCSI,3,3, ...
 ...

where:

Response Component	Description
IDref1	Beacon reference ID (primary)
IDref2	Beacon reference ID (secondary)
StationID	Specific ID number for beacon stations (appears in the last field of the GPGGA message)
name	Name of station
freq	Frequency, in kHz (scaled by 10), on which the station is transmitting. In the first line of the Example below, 2950 indicates 295.0 kHz.
lat	Scaled by 364 (+ve indicates N and -ve indicates S)
long	Longitude is scaled by 182 (+ve indicates N and -ve indicates S)
datum	1 (NAD83), 0(WGS84)
status	0 (operational), 1(undefined), 2 (no information), 3, (do not use)

Example \$PCSI,3,3,0282,0283,0891,Level Island AK,2950,20554,-24221,1,0
 \$PCSI,3,3,0306,0307,0906,Sandspit BC,3000,19377,-23991,1,0
 \$PCSI,3,3,0278,0279,0889,Annette Is. AK,3230,20044,-23951,1,0
 \$PCSI,3,3,0300,0301,0909,Alert Bay BC,3090,18412,-23099,1,0
 \$PCSI,3,3,0302,0303,0908,Amphitrite Pt BC,3150,17806,-22850,1,0
 \$PCSI,3,3,0270,0271,0885,C. Mendocino CA,2920,14718,-22641,1,0
 \$PCSI,3,3,0272,0273,0886,Fort Stevens OR,2870,16817,-22559,1,0
 \$PCSI,3,3,0304,0305,0907,Richmond BC,3200,17903,-22407,1,0
 \$PCSI,3,3,0276,0277,0888,Whidbey Is. WA,3020,17587,-22331,1,0
 ...

Additional Information

Messages (All)

Binary Messages Code

This section provides the code for the binary messages that Hemisphere GPS uses.

```
// BinaryMsg.h
#ifndef __BinaryMsg_H__
#define __BinaryMsg_H__
#ifdef __cplusplus
extern "C" {
#endif
/*
 * Copyright (c) 2006 Hemisphere GPS and CSI Wireless Inc.,
 * All Rights Reserved.
 *
 * Use and copying of this software and preparation of derivative works
 * based upon this software are permitted. Any copy of this software or
 * of any derivative work must include the above copyright notice, this
 * paragraph and the one after it. Any distribution of this software or
 * derivative works must comply with all applicable laws.
 *
 * This software is made available AS IS, and COPYRIGHT OWNERS DISCLAIMS
 * ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE, AND NOTWITHSTANDING ANY OTHER PROVISION CONTAINED HEREIN, ANY
 * LIABILITY FOR DAMAGES RESULTING FROM THE SOFTWARE OR ITS USE IS
 * EXPRESSLY DISCLAIMED, WHETHER ARISING IN CONTRACT, TORT (INCLUDING
 * NEGLIGENCE) OR STRICT LIABILITY, EVEN IF COPYRIGHT OWNERS ARE ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGES.
 */
#if defined(WIN32) || (__ARMCC_VERSION >= 300441)
    #pragma pack(push)
    #pragma pack(4)
#endif
/*****
/* SBinaryMsgHeader */
*****/
typedef struct
{
    char            m_strSOH[4];        /* start of header ($BIN) */
    unsigned short  m_byBlockID;        /* ID of message (1,2,99,98,97,96,95,94,93 or 80 ) */
    unsigned short  m_wDataLength;      /* 52 16,304,68,28,300,128,96,56, or 40 */
} SBinaryMsgHeader;
typedef struct
{
    unsigned long   ulDwordPreamble;    /* 0x4E494224 = $BIN */
    unsigned long   ulDwordInfo;        /* 0x00340001 or 0x00100002 or 0x01300063 */
} SBinaryMsgHeaderDW;
/* or 0x00440062 or 0x001C0061 or 0x012C0060 */
/* or 0x0080005F or 0x0060005E or 0x0038005D */
/* or 0x00280050 */
#define BIN_MSG_PREAMBLE    0x4E494224 /* $BIN = 0x4E494224 */
#define BIN_MSG_HEAD_TYPE1  0x00340001 /* 52 = 0x34 */
#define BIN_MSG_HEAD_TYPE2  0x00100002 /* 16 = 0x10 */
#define BIN_MSG_HEAD_TYPE99 0x01300063 /* 99 = 0x63, 304 = 0x130 */
#define BIN_MSG_HEAD_TYPE102 0x01580066 /* 102 = 0x66, 344 = 0x158 */
#define BIN_MSG_HEAD_TYPE101 0x01C00065 /* 101 = 0x65, 448 = 0x1C0 */
#define BIN_MSG_HEAD_TYPE100 0x01040064 /* 100 = 0x64, 260 = 0x104 */
#define BIN_MSG_HEAD_TYPE98  0x00440062 /* 98 = 0x62, 68 = 0x44 */
#define BIN_MSG_HEAD_TYPE97  0x001C0061 /* 97 = 0x61, 28 = 0x1C */
#define BIN_MSG_HEAD_TYPE96  0x012C0060 /* 96 = 0x60, 300 = 0x12C */
```

```

#define BIN_MSG_HEAD_TYPE95 0x0080005F /* 95 = 0x5F, 128 = 0x80 */
#define BIN_MSG_HEAD_TYPE94 0x0060005E /* 94 = 0x5E, 96 = 0x60 */
#define BIN_MSG_HEAD_TYPE93 0x0038005D /* 93 = 0x5D, 56 = 0x38 */
#define BIN_MSG_HEAD_TYPE91 0x0198005B /* 91 = 0x5B, 408 = 0x198 = total size in bytes -8 -2
-2*/
#define BIN_MSG_HEAD_TYPE89 0x00500059 /* 89 = 0x59, 80 = 0x50 */
#define BIN_MSG_HEAD_TYPE80 0x00280050 /* 80 = 0x50, 40 = 0x28 */
#define BIN_MSG_HEAD_TYPE76 0x01C0004C /* 76 = 0x4C, 448 = 0x1C0 = total size in bytes -8 -2
-2*/
#define BIN_MSG_HEAD_TYPE71 0x01C00047 /* 71 = 0x47, 448 = 0x1C0 = total size in bytes -8 -2
-2*/
#define BIN_MSG_HEAD_TYPE61 0x0140003D /* 61 = 0x3D, 320 = 0x140 */
#define BIN_MSG_HEAD_TYPE62 0x0028003E /* 62 = 0x3E, 40 = 0x28 */
#define BIN_MSG_HEAD_TYPE65 0x00440041 /* 65 = 0x41, 68 = 0x44 */
#define BIN_MSG_HEAD_TYPE66 0x01600042 /* 66 = 0x42, 352 = 0x160 */
#define BIN_MSG_HEAD_TYPE69 0x012C0045 /* 69 = 0x45, 300 = 0x12C */
#define BIN_MSG_HEAD_TYPE59 0x0100003B /* 59 = 0x3B, 256 = 0x100 */ //GPS L2C
#define BIN_MSG_HEAD_TYPE10 0x0194000A /* 10 = 0xA, 404 = 0x194 = total size in bytes -8 -2
-2*/
#if defined(_RXAIF_PLOT_MESSAGES_)
#define BIN_MSG_HEAD_TYPE11 0x0064000B /* 11 = 0x0B, 100 = 0x64 = total size(112) in
bytes -8 -2 -2*/
#endif
#define BIN_MSG_CRLF 0x0A0D /* CR LF = 0x0D, 0x0A */
#define CHANNELS_12 12
#define cBPM_SCAT_MEMSIZE 100
#if defined(_RXAIF_PLOT_MESSAGES_)
#define cBPM_AIFSCAT_MEMSIZE 16
#endif
typedef union
{
    SBinaryMsgHeader sBytes;
    SBinaryMsgHeaderDW sDWord;
} SUnionMsgHeader;
/*****
/* SBinaryMsg1 */
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char m_byAgeOfDiff; /* age of differential, seconds (255 max)*/
    unsigned char m_byNumOfSats; /* number of satellites used (12 max) */
    unsigned short m_wGPSWeek; /* GPS week */
    double m_dGPSTimeOfWeek; /* GPS tow */
    double m_dLatitude; /* Latitude degrees, -90..90 */
    double m_dLongitude; /* Latitude degrees, -180..180 */
    float m_fHeight; /* (m), Altitude ellipsoid */
    float m_fVNorth; /* Velocity north m/s */
    float m_fVEast; /* Velocity east m/s */
    float m_fVUp; /* Velocity up m/s */
    float m_fStdDevResid; /* (m), Standard Deviation of Residuals */
    unsigned short m_wNavMode;
    unsigned short m_wAgeOfDiff; /* age of diff using 16 bits */
    unsigned short m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short m_wCRLF; /* Carriage Return Line Feed */
} SBinaryMsg1; /* length = 8 + 52 + 2 + 2 = 64 */
/*****
/* SBinaryMsg2 */
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned long m_ulMaskSatsTracked; /* SATS Tracked, bit mapped 0..31 */
    unsigned long m_ulMaskSatsUsed; /* SATS Used, bit mapped 0..31 */
    unsigned short m_wGpsUtcDiff; /* GPS/UTC time difference (GPS minus UTC) */

```

```

unsigned short m_wHDOPTimes10;      /* HDOP      (0.1 units) */
unsigned short m_wVDOPTimes10;      /* VDOP      (0.1 units) */
unsigned short m_wWAASMask;         /* Bits 0-1: tracked sats, Bits 2-3:
                                     used sats, Bits 5-9 WAAS PRN 1 minus
                                     120, Bits 10-14 WAAS PRN 1 minus 120 */

unsigned short m_wChecksum;         /* sum of all bytes of the header and data */
unsigned short m_wCRLF;             /* Carriage Return Line Feed */
} SBinaryMsg2;                      /* length = 8 + 16 + 2 + 2 = 28 */
/*****
/* SChannelData
*****/
typedef struct
{
    unsigned char m_byChannel;      /* channel number */
    unsigned char m_bySV;           /* satellite being tracked, 0 == not tracked */
    unsigned char m_byStatus;       /* Status bits (code carrier bit frame...) */
    unsigned char m_byLastSubFrame; /* last subframe processed */
    unsigned char m_byEphmVFlag;    /* ephemeris valid flag */
    unsigned char m_byEphmHealth;   /* ephemeris health */
    unsigned char m_byAlmVFlag;     /* almanac valid flag */
    unsigned char m_byAlmHealth;    /* almanac health */
    char m_chElev;                  /* elevation angle */
    unsigned char m_byAzimuth;       /* 1/2 the Azimuth angle */
    unsigned char m_byURA;          /* User Range Error */
    unsigned char m_byDum;           /* Place Holder */
    unsigned short m_wCliForSNR;     /* code lock indicator for SNR divided by 32 */
    short m_nDiffCorr;              /* Differential correction * 100 */
    short m_nPosResid;              /* position residual * 10 */
    short m_nVelResid;              /* velocity residual * 10 */
    short m_nDoppHz;                /* expected doppler in HZ */
    short m_nNCOHz;                 /* track from NCO in HZ */
} SChannelData; /* 24 bytes */
/*****
/* SChannelL2Data
*****/
//#if defined(_DUAL_FREQ_)
typedef struct
{
    unsigned char m_byChannel;      /* channel number */
    unsigned char m_bySV;           /* satellite being tracked, 0 == not tracked */
    unsigned char m_byL2CX;         /* Status bits for L2P (code carrier bit frame...) */
    unsigned char m_byL1CX;         /* Status bits for L1P (code carrier bit frame...) */
    unsigned short m_wCliForSNRL2P; /* code lock indicator for SNR divided by 32 */
    unsigned short m_wCliForSNRL1P; /* code lock indicator for L1P SNR divided by 32 */
    short m_nC1_L1;                 /* C1-L1 in meters * 100 */
    short m_nP2_C1;                 /* P2-C1 in meters * 100 */
    short m_nP2_L1;                 /* P2-L1 in meters * 100 */
    short m_nL2_L1;                 /* L2-L1 in meters * 100 */
    short m_nP2_P1;                 /* P2-P1 in meters * 100 */
    short m_nNCOHz;                 /* track from NCO in HZ */
} SChannelL2Data; /* 20 bytes */
//#endif
/*****
/* SChannelL2CData for USING_GPSEL2CL
*****/
typedef struct
{
    unsigned char m_byChannel;      /* channel number */
    unsigned char m_bySV;           /* satellite being tracked, 0 == not tracked */
    unsigned char m_byL2CX;         /* Status bits for L2P (code carrier bit frame...) */
    unsigned char spare1;
    unsigned short m_wCliForSNRL2C; /* code lock indicator for SNR divided by 32 */
    unsigned short spare2;
    short m_nL2C_L1Ca;              /* L2CL - CA code error meters * 100 */
    short m_nL2C_L2P;              /* L2CL - L2P code error meters * 100 */
}

```

```

    short        m_nL2_L1;           //L2CL - L1CA phase error meters * 100
    short        m_nL2_L2P;          //L2CL - L2P phase error meters * 100
    short        spare3;
    short        m_nNCOHz;           // track from NCO in HZ
} SChannelL2CData; // 20 bytes
/*****
/* SBinaryMsg99 */
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char m_byNavMode;        /* Nav Mode FIX_NO, FIX_2D, FIX_3D (high bit
=has_diff) */
    char m_cUTCTimeDiff;              /* whole Seconds between UTC and GPS */
    unsigned short m_wGPSWeek;         /* GPS week */
    double m_dGPSTimeOfWeek;          /* GPS tow */
    SChannelData m_asChannelData[CHANNELS_12]; /* channel data */
    short m_nClockErrAtL1;            /* clock error at L1, Hz */
    unsigned short m_wSpare;           /* spare */
    unsigned short m_wChecksum;        /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;            /* Carriage Return Line Feed */
} SBinaryMsg99;                      /* length = 8 + 304 + 2 + 2 = 316 */
#define CHANNELS_SBAS_E 3
/*****
/* SBinaryMsg89 * Supports 3 SBAS Satellites */
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    long m_lGPSSecOfWeek;             /* GPS tow integer sec */
    unsigned char m_byMaskSBASTracked; /* SBAS Sats Tracked, bit mapped 0..3 */
    unsigned char m_byMaskSBASUSED;    /* SBAS Sats Used, bit mapped 0..3 */
    unsigned short m_wSpare;           /* spare */
    SChannelData m_asChannelData[CHANNELS_SBAS_E]; /* SBAS channel data */
    unsigned short m_wChecksum;        /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;            /* Carriage Return Line Feed */
} SBinaryMsg89;                      /* length = 8 + 80 + 2 + 2 = 92 */
/*****
/* SBinaryMsg100 */
*****/
// #if defined(_DUAL_FREQ_)
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char m_byNavMode;        /* Nav Mode FIX_NO, FIX_2D, FIX_3D (high bit
=has_diff) */
    char m_cUTCTimeDiff;              /* whole Seconds between UTC and GPS */
    unsigned short m_wGPSWeek;         /* GPS week */
    unsigned long m_ulMaskSatsUsedL2P; /* L2P SATS Used, bit mapped 0..31 */
    double m_dGPSTimeOfWeek;          /* GPS tow */
    unsigned long m_ulMaskSatsUsedL1P; /* L1P SATS Used, bit mapped 0..31 */
    SChannelL2Data m_asChannelData[CHANNELS_12]; /* channel data */
    unsigned short m_wChecksum;        /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;            /* Carriage Return Line Feed */
} SBinaryMsg100;                     /* length = 8 + 260 + 2 + 2 = 272 */
// #endif
/*****
/* SBinaryMsg59 for USING_GPS_L2CL */
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char m_byNavMode;        /* Nav Mode FIX_NO, FIX_2D, FIX_3D (high bit
=has_diff) */ // 1 byte

```

```

char          m_cUTCTimeDiff;      /* whole Seconds between UTC and GPS */
                                //1 byte
unsigned short m_wGPSWeek;         /* GPS week */
                                //2 bytes
unsigned long  m_ulMaskSatsUsedL2P; /* L2P SATS Used, bit mapped 0..31 */
                                //4 bytes
double        m_dGPSTimeOfWeek;   /* GPS tow */
                                //8 bytes
SChannelL2CData m_asChannelData[CHANNELS_12]; /* channel data */
                                //20*12 bytes
unsigned short m_wChecksum;        /* sum of all bytes of the header and data */
unsigned short m_wCRLF;            /* Carriage Return Line Feed */
} SBinaryMsg59;                   /* length = 8 + 260 + 2 + 2 = 272 */
/*****
/*  SSVALmanData
*****/
typedef struct
{
    short          m_nDoppHz;      /* doppler in HZ for stationary receiver */
    unsigned char  m_byCountUpdate; /* count of almanac updates */
    unsigned char  m_bySVindex;    /* 0 through 31 (groups of 8)*/
    unsigned char  m_byAlmVFlag;   /* almanac valid flag */
    unsigned char  m_byAlmHealth;  /* almanac health */
    char          m_chElev;        /* elevation angle */
    unsigned char  m_byAzimuth;    /* 1/2 the Azimuth angle */
} SSVALmanData; /* 8 bytes */
/*****
/*  SBinaryMsg98
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    SSVALmanData    m_asAlmanData[8]; /* SV data, 8 at a time */
    unsigned char  m_byLastAlman;    /* last almanac processed */
    unsigned char  m_byIonoUTCFlag;  /* iono UTC flag */
    unsigned short m_wSpare;         /* spare */
    unsigned short m_wChecksum;      /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg98; /* length = 8 + (64+1+1+2) + 2 + 2 = 80 */
/*****
/*  SBinaryMsg97
*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned long  m_ulCPUFactor;    /* CPU utilization Factor (%=multby 450e-6) */
    unsigned short m_wMissedSubFrame; /* missed subframes */
    unsigned short m_wMaxSubFramePend; /* max subframe pending */
    unsigned short m_wMissedAccum;    /* missed accumulations */
    unsigned short m_wMissedMeas;     /* missed measurements */
    unsigned long  m_ulSpare1;        /* spare 1 (zero)*/
    unsigned long  m_ulSpare2;        /* spare 2 (zero)*/
    unsigned long  m_ulSpare3;        /* spare 3 (zero)*/
    unsigned short m_wSpare4;         /* spare 4 (zero)*/
    unsigned short m_wSpare5;         /* spare 5 (zero)*/
    unsigned short m_wChecksum;      /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg97; /* length = 8 + (28) + 2 + 2 = 40 */
/*****
/*  SObservations
*****/
typedef struct
{
    unsigned long  m_ulCS_TT_SNR_PRN; /* Bits 0-7 PRN (PRN is 0 if no data) */
                                /* Bits 8-15 SNR_value */

```



```

        SNR = 10.0*log10( 0.8192*SNR_value) */
/* Bits 16-23 Phase Track Time in units
   of 1/10 second (range = 0 to 25.5
   seconds (see next word) */
/* Bits 24-31 Cycle Slip Counter
   Increments by 1 every cycle slip
   with natural roll over after 255 */
unsigned long    m_ulDoppler_FL; /* Bit  0: 1 if Valid Phase, 0 otherwise
                                   Bit  1: 1 if Track Time > 25.5 sec,
                                   0 otherwise
                                   Bits 2-3: unused
                                   Bits 4-32: Signed (two's compliment)
                                   doppler in units of m/sec x 4096.
                                   (i.e., LSB = 1/4096). Range =
                                   +/- 32768 m/sec. Computed as
                                   phase change over 1/10 sec. */

double          m_dPseudoRange; /* pseudo ranges (m) */
double          m_dPhase;        /* phase (m) L1 wave len = 0.190293672798365*/
} SObservations; /* 24 bytes */
/*****
/* SBinaryMsg96
/*****
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short  m_wSpare1; /* spare 1 (zero)*/
    unsigned short  m_wWeek;   /* GPS Week Number */
    double          m_dTow;    /* Predicted GPS Time in seconds */
    SObservations  m_asObvs[CHANNELS_12]; /* 12 sets of observations */
    unsigned short  m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;    /* Carriage Return Line Feed */
} SBinaryMsg96; /* length = 8 + (300) + 2 + 2 = 312 */
/*****
/* SBinaryMsg95
/*****
/* sent only upon command or when values change */
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short  m_wSV; /* The satellite to which this data belongs. */
    unsigned short  m_wSpare1; /* spare 1 (chan number (as zero 9/1/2004)*/
    unsigned long   m_TOW6SecOfWeek; /* time at which this arrived (LSB = 6sec) */
    unsigned long   m_SF1words[10]; /* Unparsed SF 1 message words. */
    unsigned long   m_SF2words[10]; /* Unparsed SF 2 message words. */
    unsigned long   m_SF3words[10]; /* Unparsed SF 3 message words. */
    /* Each of the subframe words contains
       one 30-bit GPS word in the lower
       30 bits, The upper two bits are ignored
       Bits are placed in the words from left to
       right as they are received */
    unsigned short  m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;    /* Carriage Return Line Feed */
} SBinaryMsg95; /* length = 8 + (128) + 2 + 2 = 140 */
/*****
/* SBinaryMsg94
/*****
/* sent only upon command or when values change */
typedef struct
{
    SUnionMsgHeader m_sHead;
    /* Iono parameters. */
    double          m_a0,m_a1,m_a2,m_a3; /* AFCRL alpha parameters. */
    double          m_b0,m_b1,m_b2,m_b3; /* AFCRL beta parameters. */
    /* UTC conversion parameters. */
    double          m_A0,m_A1; /* Coeffs for determining UTC time. */

```

```

    unsigned long  m_tot;                /* Reference time for A0 & A1, sec of GPS week. */
    unsigned short m_wnt;                /* Current UTC reference week number. */
    unsigned short m_wnlsf;              /* Week number when dtlsf becomes effective. */
    unsigned short m_dn;                 /* Day of week (1-7) when dtlsf becomes effective.
*/
    short          m_dtls;               /* Cumulative past leap seconds. */
    short          m_dtlsf;              /* Scheduled future leap seconds. */
    unsigned short m_wSpare1;            /* spare 4 (zero)*/
    unsigned short m_wChecksum;          /* sum of all bytes of the header and data */
    unsigned short m_wCRLF;              /* Carriage Return Line Feed */
} SBinaryMsg94;                         /* length = 8 + (96) + 2 + 2 = 108 */
/*****
/* SBinaryMsg93
/*****
/* sent only upon command or when values change */
/* WAAS ephemeris */
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short  m_wSV;               /* The satellite to which this data belongs. */
    unsigned short  m_wWeek;             /* Week corresponding to m_lTOW*/
    unsigned long   m_lSecOfWeekArrived; /* time at which this arrived (LSB = 1sec) */
    unsigned short  m_wIODE;
    unsigned short  m_wURA;             /* See 2.5.3 of Global Pos Sys Std Pos Service Spec
*/
    long m_lTOW;                         /* Sec of WEEK Bit0 = 1 sec */
    long m_lXG;                          /* Bit 0 = 0.08 m */
    long m_lYG;                          /* Bit 0 = 0.08 m */
    long m_lZG;                          /* Bit 0 = 0.4 m */
    long m_lXGDot;                       /* Bit 0 = 0.000625 m/sec */
    long m_lYGDot;                       /* Bit 0 = 0.000625 m/sec */
    long m_lZGDot;                       /* Bit 0 = 0.004 m/sec */
    long m_lXGDotDot;                   /* Bit 0 = 0.0000125 m/sec/sec */
    long m_lYGDotDot;                   /* Bit 0 = 0.0000125 m/sec/sec */
    long m_lZGDotDot;                   /* Bit 0 = 0.0000625 m/sec/sec */
    short m_nGf0;                       /* Bit 0 = 2**-31 sec */
    short m_nGf0Dot;                    /* Bit 0 = 2**-40 sec/sec */
    unsigned short  m_wChecksum;          /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;              /* Carriage Return Line Feed */
} SBinaryMsg93;                         /* length = 8 + (56) + 2 + 2 = 68 */
/*****
/* SBinaryMsg80
/*****
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short  m_wPRN;               /* Broadcast PRN */
    unsigned short  m_wSpare;             /* spare (zero) */
    unsigned long   m_ulMsgSecOfWeek;     /* Seconds of Week For Message */
    unsigned long   m_aulWaasMsg[8];      /* Actual 250 bit waas message*/
    unsigned short  m_wChecksum;          /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;              /* Carriage Return Line Feed */
} SBinaryMsg80;                         /* length = 8 + (40) + 2 + 2 = 52 */
/*****
/* SMsg91Data
/*****
typedef struct
{
    unsigned char  bySV;                  /* satellite being tracked, 0 == not tracked */
    unsigned char  byStatus;              /* Status bits (code carrier bit frame...) */
    unsigned char  byStatusSlave;         /* Status bits (code carrier bit frame...) */
    unsigned char  byChannel;             /* Not used */

    unsigned short wEpochSlew;            /* 20*_20MS_EPOCH_SLEW + _1MS_EPOCH_SLEW */
    unsigned short wEpochCount;          /* epoch_count */

```

```

    unsigned long   codeph_SNR;                /* 0-20 = code phase (21 bits), 28-32 =
SNR/4096, upper 4 bits */
    unsigned long   ulCarrierCycles_SNR;        /* 0-23 = carrier cycles, 24-32 = SNR/4096
lower 8 bits */
    unsigned short  wDCOPhaseB10_HalfWarns;     /* 0-11 = DCO phase, 12-14 = Half Cycle Warn
15 = half Cycle added */
    unsigned short  m_wPotentialSlipCount;      /* potential slip count */
/* SLAVE DATA */
    unsigned long   codeph_SNR_Slave;           /* 0-20 = code phase (21 bits), 28-32 =
SNR/4096, upper 4 bits */
    unsigned long   ulCarrierCycles_SNR_Slave;  /* 0-23 = carrier cycles, 24-32 = SNR/4096
lower 8 bits */
    unsigned short  wDCOPhaseB10_HalfWarns_Slave; /* 0-11 = DCO phase, 12-14 = Half Cycle Warn
15 = half Cycle added */
    unsigned short  m_wPotentialSlipCount_Slave; /* potential slip count */
} SMsg91Data; /* 32 bytes */
/*****
/* SBinaryMsg91
/* Comment: Transmits data from Takemeas.c
/* debugging structure.
/* Added by bbadke 7/07/2003
*****/

typedef struct
{
    SUnionMsgHeader  m_sHead;                /* 8 */
    double           m_sec;                  /* 8 bytes */
    int              m_iWeek;                /* 4 bytes */
    unsigned long    m_Tic;                  /* 4 bytes */
    long             lTicOfWeek;              /* 4 bytes */
    long             lProgTic;                /* 4 bytes */
    SMsg91Data       s91Data[CHANNELS_12];   /* 12*32= 384 bytes */
    unsigned short   m_wChecksum;            /* sum of all bytes of the header and data */
    unsigned short   m_wCRLF;                /* Carriage Return Line Feed */
} SBinaryMsg91; /* length = 8 + (408) + 2 + 2 = 420 */
/*****
/* SObsPacket
*****/

typedef struct
{
    unsigned long    m_ulCS_TT_W3_SNR;        /* Bits 0-11 (12 bits) =SNR_value
For L1 SNR = 10.0*log10( 0.1024*SNR_value)
FOR L2 SNR = 10.0*log10( 0.1164*SNR_value) */
/* Bits 12-14 (3 bits) = 3 bits of warning
for potential 1/2 cycle slips. A warning
exists if any of these bits are set. */
/* bit 15: (1 bit) 1 if Track Time > 25.5 sec,
0 otherwise */
/* Bits 16-23 (8 bits): Track Time in units
of 1/10 second (range = 0 to 25.5 seconds) */
/* Bits 24-31 (8 bits) = Cycle Slip Counter
Increments by 1 every cycle slip
with natural roll-over after 255 */
    unsigned long    m_ulP7_Doppler_FL;      /* Bit 0: (1 bit) 1 if Valid Phase, 0 otherwise
Bit 1-23: (23 bits) =Magnitude of doppler
LSB = 1/512 cycle/sec
Range = 0 to 16384 cycle/sec
Bit 24: sign of doppler, 1=negative, 0=pos
Bits 25-31 (7 bits) = upper 7 bits of the
23 bit carrier phase.
LSB = 64 cycles, MSB = 4096 cycles */
    unsigned long    m_ulCodeAndPhase;       /* Bit 0-15 (16 bits) lower 16 bits of code
pseudorange
LSB = 1/256 meters
MSB = 128 meters
Note, the upper 19 bits are given in

```

```

        m_aulCACodeMSBsPRN[] for CA code
        Bit 16-31 lower 16 bits of the carrier phase,
            7 more bits are in m_ulP7_Doppler_FL
        LSB = 1/1024 cycles
        MSB = 32 cycles */

} SObsPacket; /* 12 bytes , note: all zero if data not available */
/* A NOTE ON DECODING MESSAGE 76
 * Notation: "code" -- is taken to mean the PseudoRange derived from code phase.
 *           "phase" -- is taken to mean range derived from carrier phase.
 *           This will contain cycle ambiguities.
 *
 * Only the lower 16 bits of L1P code, L2P code and the lower 23 bits of
 * carrier phase are provided. The upper 19 bits of the L1CA code are found
 * in m_aulCACodeMSBsPRN[]. The upper 19 bits of L1P or L2P must be derived
 * using the fact that L1P and L2P are within 128 meters of L1CA. To
 * determine L1P or L2P, use the lower 16 bits provided in the message and
 * set the upper bits to that of L1CA. Then add or subtract one LSB of the
 * upper bits (256 meters) so that L1P or L2P are within 1/2 LSB (128 meters)
 * of the L1CA code.
 * The carrier phase is in units of cycles, rather than meters,
 * and is held to within 1023 cycles of the respective code range. Only
 * the lower 16+7=23 bits of carrier phase are transmitted in Msg 76.
 * In order to determine the remaining bits, first convert the respective
 * code range (determined above) into cycles by dividing by the carrier
 * wavelength. Call this the "nominal reference phase". Next extract the 16
 * and 7 bit blocks of carrier phase from Msg 76 and arrange to form the lower
 * 23 bits of carrier phase. Set the upper bits (bit 23 and above) equal to
 * those of the nominal reference phase. Then, similar to what was done for
 * L1P and L2P, add or subtract the least significant upper bit (8192 cycles)
 * so that carrier phase most closely agrees with the nominal reference phase
 * (to within 4096 cycles).
 */
#define CHANNELS_12_PLUS (CHANNELS_12+2) /* up to two SBAS satellites */
#define CHANNELS_L1_E (CHANNELS_12+CHANNELS_SBAS_E) /* All L1 (including SBAS satellites)
 */
/*****
 */
/* SBinaryMsg76 */
/*****
 */
typedef struct
{
    SUnionMsgHeader m_sHead;
    double m_dTow; /* GPS Time in seconds */
    unsigned short m_wWeek; /* GPS Week Number */
    unsigned short m_wSpare1; /* spare 1 (zero)*/
    unsigned long m_ulSpare2; /* spare 2 (zero)*/
    SObsPacket m_asL2PObs[CHANNELS_12]; /* 12 sets of L2(P) observations */
    SObsPacket m_asL1CAObs[CHANNELS_L1_E]; /* 15 sets of L1(CA) observations */
    unsigned long m_aulCACodeMSBsPRN[CHANNELS_L1_E]; /* array of 15 words.
        bit 7:0 (8 bits) = satellite PRN, 0
        if no satellite
        bit 12:8 (5 bits) = spare
        bit 31:13 (19 bits) = upper 19 bits
        of L1CA LSB = 256 meters
        MSB = 67108864 meters */
    unsigned long m_auL1Pword[CHANNELS_12]; /* array of 12 words relating to L1(P) code.
        Bit 0-15 (16 bits) lower 16 bits of the
        L1P code pseudo range.
        LSB = 1/256 meters
        MSB = 128 meters
        Bits 16-27 (12 bits) = L1P SNR_value
        SNR = 10.0*log10( 0.1164*SNR_value)
        If Bits 16-27 all zero, no L1P track
        Bits 28-31 (4 bits) spare */
    unsigned short m_wChecksum; /* sum of all bytes of the header and data */

```

```

    unsigned short    m_wCRLF;                /* Carriage Return Line Feed */
} SBinaryMsg76;                               /* length = 8 + (448) + 2 + 2 = 460 */
/*****
/*   SMsg71DataL1
*****/
typedef struct
{
    unsigned char    bySV;                    /* satellite being tracked, 0 == not tracked */
    unsigned char    byStatus;                /* Status bits (code carrier bit frame...) */
    unsigned char    byStatusL1P;            /* 0-8 lower 8 bits of L1P SNR/32768, if zero and
                                              if upper two bits of m_wSNR_codeph_L1P are
zero
                                              then L1P is not tracking */
    unsigned char    byStatusL2P;            /* Status bits (code carrier phase ...) */
    unsigned short   wEpochSlew;            /* 20*_20MS_EPOCH_SLEW + _1MS_EPOCH_SLEW */
    unsigned short   wEpochCount;          /* epoch_count */
    unsigned long    codeph_SNR;            /* 0-20 = code phase (21 bits), 28-32 = SNR/4096,
upper 4 bits */
    unsigned long    ulCarrierCycles_SNR;    /* 0-23 = carrier cycles, 24-32 = SNR/4096 lower
8 bits */
    unsigned short   wDCOPhaseB10_HalfWarns; /* 0-11 = DCO phase, 12-14 = Half Cycle Warn
                                              15 = half Cycle added */
    unsigned short   m_wPotentialSlipCount; /* potential slip count */
} SMsg71DataL1; /* 20 bytes */
/*****
/*   SMsg71DataL1PL2P
*****/
typedef struct
{
    /* L1P and L2P Data */
    // unsigned long codeph_SNR_L1P; NOT USED YET /* 0-22 = L1 code phase (23 bits), 28-32 =
SNR/8192, upper 4 bits */
    unsigned long    codeph_SNR_L2P;        /* 0-22 = L2P code phase (23 bits), 28-32
= SNR/8192, upper 4 bits */
    unsigned long    ulCarrierCycles_SNR_L2P; /* 0-23 = carrier cycles, 24-32 = SNR/8192
lower 8 bits */
    unsigned short   wDCOPhaseB10_L2P;      /* 0-11 = DCO phase, 12-15 = Spare */
    unsigned short   m_wSNR_codeph_L1P;    /* 0-13 = lower 14 bits of L1P code, 14-
15 SNR/32768 Upper 2 bits */
                                              /* To get full L1P code, use upper bits
form L2P and adjust by
                                              +/- 2**14 if necessary */
} SMsg71DataL1PL2P; /* 12 bytes */
/*****
/*   SBinaryMsg71
/*   Comment: Transmits data from Takemeas.c
/*   debugging structure for Dual Freq.
*****/
typedef struct
{
    SUnionMsgHeader  m_sHead;                /* 8 */
    double           m_sec;                  /* 8 bytes */
    int              m_iWeek;                /* 4 bytes */
    unsigned long    m_Tic;                  /* 4 bytes */
    long             lTicOfWeek;             /* 4 bytes */
    long             lProgTic;               /* 4 bytes */
    SMsg71DataL1PL2P s91L2PData[CHANNELS_12]; /* 12*12 = 144 bytes */
    SMsg71DataL1     s91Data[CHANNELS_12_PLUS]; /* 14*20 = 280 bytes */
    unsigned short    m_wChecksum;           /* sum of all bytes of the header and data
*/
    unsigned short    m_wCRLF;                /* Carriage Return Line Feed */
} SBinaryMsg71;                               /* length = 8 + (448) + 2 + 2 = 460 */
// SBinaryMsg10
// Comment: Transmits scatter plot data from

```

```

//          buffacc.c
//
////////////////////////////////////
enum eBIN10_TYPE {eBIN10_GPSSL1CA=0,eBIN10_GPSSL1P,eBIN10_GPSSL2P,
                  eBIN10_GLONASSL1,eBIN10_GLONASSL2,eBIN10_GPSSL2CL,eBIN10_GPSSL5Q};
typedef struct
{
    SUnionMsgHeader m_sHead;           // 8 bytes
    unsigned short m_awScatterPlotDataI[cBPM_SCAT_MEMSIZE]; //100*2 = 200 bytes
    unsigned short m_awScatterPlotDataQ[cBPM_SCAT_MEMSIZE]; //100*2 = 200 bytes
    unsigned short m_wChannel;
    unsigned short m_wSigType;         // one of eBIN10_TYPE
    unsigned short m_wChecksum;        // sum of all bytes of the header and data
    unsigned short m_wCRLF;            // Carriage Return Line Feed
} SBinaryMsg10;                       // length = 8 +200 +200 +2 +2 +2 +2 = 416
#ifdef _RXAIF_PLOT_MESSAGES_
////////////////////////////////////
// SBinaryMsg11
// Comment: Transmits scatter plot data for RXGNSS_AIF statistics
//
////////////////////////////////////
enum eBIN11_TYPE {eBIN11_COUNTS=0,eBIN11_VALUES};
typedef struct
{
    SUnionMsgHeader m_sHead;           // 8 bytes
    unsigned short m_awScatterPlotDataValues[cBPM_AIFSCAT_MEMSIZE]; //16*2 = 32 bytes
    unsigned short m_awScatterPlotDataCntMag[cBPM_AIFSCAT_MEMSIZE]; //16*2 = 32 bytes
    unsigned short m_awScatterPlotDataCntDCoff[cBPM_AIFSCAT_MEMSIZE]; //16*2 = 32 bytes
    unsigned short m_wChannel;         // aif_sel 0: AIF_A, 1: AIF_B, ...
    unsigned short m_wSigType;         // one of eBIN11_TYPE
    unsigned short m_wChecksum;        // sum of all bytes of the header and data
    unsigned short m_wCRLF;            // Carriage Return Line Feed
} SBinaryMsg11;                       // length = 8 +32 +32 +32 +2 +2 +2 +2 = 112

#endif
/*****
/* SGLONASSChanData */
*****/
typedef struct
{
    unsigned char m_bySV;              /* Bit (0-6) = SV slot, 0 == not tracked
    * Bit 7 = Knum flag
    * = KNum+8 if bit 7 set
    */

    unsigned char m_byAlm_Ephm_Flags; /* ephemeris and almanac status flags */
    /* bit 0: Ephemeris available but timed out
    * bit 1: Ephemeris valid
    * bit 2: Ephemeris health OK
    * bit 3: unused
    * bit 4: Almanac available
    * bit 5: Almanac health OK
    * bit 6: unused
    * bit 7: Satellite doesn't exist
    */

    unsigned char m_byStatus_L1;       /* Status bits (code carrier bit frame...) */
    unsigned char m_byStatus_L2;       /* Status bits (code carrier bit frame...) */
    char m_chElev;                     /* elevation angle */
    unsigned char m_byAzimuth;          /* 1/2 the Azimuth angle */
    unsigned char m_byLastMessage;      /* last message processed */
    unsigned char m_bySlip01;          /* cycle slip on chan 1 */
    unsigned short m_wCliForSNR_L1;     /* code lock indicator for SNR divided by 32 */
    unsigned short m_wCliForSNR_L2;     /* code lock indicator for SNR divided by 32 */
    short m_nDiffCorr_L1;              /* Differential correction * 100 */
    short m_nDoppHz;                   /* expected doppler in HZ at glonass L1 */
    short m_nNCOHz_L1;                 /* track from NCO in HZ */

```

```

    short          m_nNCOHz_L2;          /* track from NCO in HZ */
    short          m_nPosResid_1;        /* position residual 1 * 1000 */
    short          m_nPosResid_2;        /* position residual 2 * 1000 */
} SGLONASSChanData; /* 24 bytes */
/*****/
/* SBinaryMsg69 */
/*****/
typedef struct
{
    SUnionMsgHeader m_sHead;
    long            m_lSecOfWeek;        /* tow */
    unsigned short  m_wL1usedNavMask;    /* mask of L1 channels used in nav solution */
    unsigned short  m_wL2usedNavMask;    /* mask of L2 channels used in nav solution */
    SGLONASSChanData m_asChannelData[CHANNELS_12]; /* channel data 12X24 = 288 */
    unsigned short  m_wWeek;            /* week */
    unsigned char    m_bySpare01;        /* spare 1 */
    unsigned char    m_bySpare02;        /* spare 2 */
    unsigned short  m_wChecksum;         /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;            /* Carriage Return Line Feed */
} SBinaryMsg69; /* length = 8 + 300 + 2 + 2 = 312 */
/*****/
/* SMsg61Data */
/*****/
typedef struct
{
    unsigned char  bySV;                /* satellite slot 0== not tracked */
    unsigned char  byStatusL1;          /* Status bits (code carrier bit frame...) */
    unsigned char  byStatusL2;          /* Status bits (code carrier bit frame...) */
    unsigned char  byL1_L2_DCO;         /* 0-3 = upper 4 bits of L1 carrier DCO Phase
                                         * 4-7 = upper 4 bits of L2 carrier DCO Phase
                                         */
    unsigned short wEpochSlewL1;        /* 0-9 = slew, 0 to 1000 count for ms of sec
                                         * 10-15 = 6 bits of L1 slip count */
    unsigned short wEpochCountL1;       /* 0-9 = epoch_count, 0 to 1000 count for ms of sec
                                         * 10-15 = 6 bits of L2 slip count */
    unsigned long  codeph_SNR_L1;        /* 0-20 = L1 code phase (21 bits = 9+12),
                                         * 21-32 = L1 SNR/4096 (upper 11 of 12 bits) */
    unsigned long  ulCarrierCycles_L1;   /* 0-23 = L1 carrier cycles,
                                         * 24-32 = L1 Carrier DCO lower 8 bits */
    unsigned long  codeph_SNR_L2;        /* 0-20 = L2 code phase (21 bits = 9+12),
                                         * 21-32 = L2 SNR/4096 (upper 11 of 12 bits) */
    unsigned long  ulCarrierCycles_L2;   /* 0-23 = L2 carrier cycles,
                                         * 24-32 = L2 Carrier DCO lower 8 bits */
} SMsg61Data; /* 24 bytes */
/*****/
/* SBinaryMsg61 */
/* Comment: Transmits data from TakemeasGLONASS.c */
/* debugging structure for Dual Freq. */
/*****/
typedef struct
{
    SUnionMsgHeader m_sHead;            /* 8 */
    unsigned long    m_Tic;              /* 4 bytes */
    unsigned long    ulSpare;            /* 4 bytes */
    unsigned short   awHalfWarns[CHANNELS_12]; /* 12*2 = 24 bytes */
                                         /* each word is
                                         * bit 0-2 L1 Half Cycle Warn
                                         * bit 3 = L1 half cycle added
                                         * bit 4-6 L2 Half Cycle Warn
                                         * bit 7 = L2 half cycle added
                                         * 8 = LSB of 12 bit L1 SNR/4096
                                         * 9 = LSB of 12 bit L2 SNR/4096
                                         * bit 10-15 Ktag of the SV */
    SMsg61Data       as61Data[CHANNELS_12]; /* 12*24 = 288 bytes */
    unsigned short    m_wChecksum;        /* sum of all bytes of the header and data */
}

```

```

    unsigned short    m_wCRLF;                /* Carriage Return Line Feed */
} SBinaryMsg61;                                /* length = 8 + (320) + 2 + 2 = 332 */
/*****/
/* SBinaryMsg66 GLONASS OBS (see notes on message 76) */
/*****/
typedef struct
{
    SUnionMsgHeader    m_sHead;
    double              m_dTow;                /* Time in seconds */
    unsigned short      m_wWeek;               /* GPS Week Number */
    unsigned short      m_wSpare1;             /* spare 1 (zero) */
    unsigned long       m_ulSpare2;            /* spare 2 (zero) */
    SObsPacket          m_asL1Obs[CHANNELS_12]; /* 12 sets of L1(Glonass) observations */
    SObsPacket          m_asL2Obs[CHANNELS_12]; /* 12 sets of L2(Glonass) observations */
    unsigned long       m_aulL1CodeMSBsSlot[CHANNELS_12]; /* array of 12 words.
                                                    bit 7:0 (8 bits) = satellite Slot, 0
                                                    if no satellite
                                                    bit 12:8 (5 bits) = spare
                                                    bit 31:13 (19 bits) = upper 19 bits
                                                    of L1  LSB = 256 meters
                                                    MSB = 67108864 meters */
    unsigned short      m_wChecksum;           /* sum of all bytes of the header and data */
    unsigned short      m_wCRLF;               /* Carriage Return Line Feed */
} SBinaryMsg66;                                /* length = 8 + (352) + 2 + 2 = 364 */
/*****/
/* SGLONASS_String, added for glonass strings */
/*****/
typedef struct
{
    unsigned long m_aul85Bits[3]; /* holds bits 9-85 of the GLONASS string */
    /*
    * bit order in message 65
    *
    * MSB                                LSB
    * m_aul85Bits[0]: 85 84.....54
    * m_aul85Bits[1]: 53 52.....22
    * m_aul85Bits[2]: 21 20.....9
    */
} SGLONASS_String; /* 12 bytes (max of 96 bits) */
/*****/
/* SBinaryMsg65, added by JL for glonass subframe immediate data + string_5 */
/*****/
/* sent only upon command or when values change (not including changes in tk) */
typedef struct
{
    SUnionMsgHeader    m_sHead;
    unsigned char       m_bySV;                /* The satellite to which this data
belongs. */
    unsigned char       m_byKtag;              /* The satellite K Number + 8. */
    unsigned short      m_wSpare1;             /* Spare, keeps alignment to 4 bytes */
    unsigned long       m_ulTimeReceivedInSeconds; /* time at which this arrived */
    SGLONASS_String     m_asStrings[5];        /* first 5 Strings of Glonass Frame (60
bytes) */
    unsigned short      m_wChecksum;           /* sum of all bytes of the header and
data */
    unsigned short      m_wCRLF;               /* Carriage Return Line Feed */
} SBinaryMsg65;                                /* length = 8 + (68) + 2 + 2 = 80 */
/*****/
/* SBinaryMsg62, Glonass almanac data. Containing string
* 5 and the two string pair for each satellite after string 5.
* String 5 contains the time reference for the glonass almanac
* and gps-glonass time differences.
*
*****/
typedef struct
{

```

```

    SUnionMsgHeader m_sHead;
    unsigned char   m_bySV;          /* The satellite to which this data
belongs. */
    unsigned char   m_byKtag_ch;     /* Proprietary data */
    unsigned short  m_wSpare1;       /* Spare, keeps alignment to 4 bytes */
    SGLONASS_String m_asStrings[3];  /* glonass almanac data (36 bytes)
                                     0 & 1 = Two almanac SFs, 3= SF 5*/
    unsigned short  m_wChecksum;     /* sum of all bytes of the header and
data */
    unsigned short  m_wCRLF;         /* Carriage Return Line Feed */
} SBinaryMsg62;                    /* length = 8 + (40) + 2 + 2 = 52 */
#if defined(WIN32) || (__ARMCC_VERSION >= 300441)
    #pragma pack(pop)
#endif
#ifdef __cplusplus
}
#endif
#endif // __BinaryMsg_H_

```

Bin1 Message

Message Type [Binary](#)

Description GPS position message (position and velocity data)

Command Format to Request Message \$JBIN,1,R<CR><LF>

where:

- '1' = Bin1 message
- 'R' = message rate in Hz (20, 10, 2, 1, 0, or .2)

Message Format

Component	Description	Type	Bytes	Values
AgeOfDiff	Age of differential, seconds. Use Extended AgeOfDiff first. If both = 0, then no differential	Byte	1	0 to 255
NumOfSats	Number of satellites used in the GPS solution	Byte	1	0 to 12
GPSWeek	GPS week associated with this message	Unsigned short	2	0 to 65536
GPSTimeOf Week	GPS tow (sec) associated with this message	Double	8	0.0 to 604800.0
Latitude	Latitude in degrees north	Double	8	-90.0 to 90.0
Longitude	Longitude in degrees East	Double	8	-180.0 to 180.0
Height	Altitude above the ellipsoid in meters	Float	4	
VNorth	Velocity north in m/s	Float	4	
VEast	Velocity east in n/s	Float	4	
Vup	Velocity up in m/s	Float	4	
StdDevResid	Standard deviation of residuals in meters	Float	4	Positive

NavMode	Navigation mode: 0 = No fix 1 = FIX_2D 2 = FIX_3D (or FIX_3d and solving ambiguities if rover) 3 = FIX_2D and Diff 4 = FIX_3D Diff (not solving ambiguities if rover) 5 = RTK Search 6 = FIX_3D and Diff and RTK solution If bit 7 is set (left-most bit), then this is a manual position	Unsigned short	2	Bits 0 through 6 = Navmode Bit 7 = Manual mark
Extended AgeOfDiff	Extended age of differential, seconds. If 0, use 1 byte AgeOfDiff listed above	Unsigned short	2	0 to 65536

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char   m_byAgeOfDiff;    /* age of differential, seconds (255 max) */
    unsigned char   m_byNumOfSats;    /* number of satellites used (12 max) */
    unsigned short  m_wGPSWeek;       /* GPS week */
    double          m_dGPSTimeOfWeek; /* GPS tow */
    double          m_dLatitude;      /* Latitude degrees, -90..90 */
    double          m_dLongitude;     /* Longitude degrees, -180..180 */
    float           m_fHeight;        /* (m), Altitude ellipsoid */
    float           m_fVNorth;        /* Velocity north m/s */
    float           m_fVEast;         /* Velocity east m/s */
    float           m_fVUp;           /* Velocity up m/s */
    float           m_fStdDevResid;   /* (m), Standard Deviation of Residuals */
    unsigned short  m_wNavMode;
    unsigned short  m_wAgeOfDiff;     /* age of diff using 16 bits */
    unsigned short  m_wChecksum;      /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg1; /* length = 8 + 52 + 2 + 2 = 64 */
```

Additional Information

Message has a BlockID of 1 and is 52 bytes, excluding the header and epilogue

Related Commands

[JBIN](#)

Bin2 Message

Message Type [Binary](#)

Description GPS DOPs (Dilution of Precision)

This message contains various quantities that are related to the GPS solution.

Command Format to Request Message \$JBIN, 2, R<CR><LF>

where:

- '2' = Bin2 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
MaskSatsTracked	Mask of satellites tracked by the GPS. Bit 0 corresponds to the GPS satellite with PRN 1.	Unsigned long	4	Individual bits represent satellites
MaskSatsUsed	Mask of satellites used in the GPS solution. Bit 0 corresponds to the GPS satellite with PRN 1.	Unsigned long	4	Individual bits represent satellites
GpsUtcDiff	Whole seconds between UTC and GPS time (GPS minus UTC)	Unsigned short	2	Positive
HDOPTimes10	Horizontal dilution of precision scaled by 10 (0.1 units)	Unsigned short	2	Positive
VDOPTimes10	Vertical dilution of precision scaled by 10 (0.1 units)	Unsigned short	2	Positive
WAASMask	PRN and tracked or used status masks	Unsigned short	2	See following
<ul style="list-style-type: none"> • Bit 00 - Mask of satellites tracked by first WAAS satellite • Bit 01 - Mask of satellites tracked by second WAAS satellite • Bit 02 - Mask of satellites used by first WAAS satellite • Bit 03 - Mask of satellites used by second WAAS satellite • Bit 04 - Unused • Bits 05-09 - Value used to find PRN of first WAAS satellite (This value + 120 = PRN) • Bits 10-14 - Value used to find PRN of second WAAS satellite (This value + 120 = PRN) • Bit 15 - Unused 				

Structure

```
typedef struct
{
    SUnionMsgHeader  m_sHead;
    unsigned long    m_ulMaskSatsTracked; /* SATS Tracked, bit mapped 0..31 */
    unsigned long    m_ulMaskSatsUsed;   /* SATS Used, bit mapped 0..31 */
}
```

```
    unsigned short    m_wGpsUtcDiff;      /* GPS/UTC time difference (GPS minus UTC) */
    unsigned short    m_wHDOPtimes10;    /* HDOP (0.1 units) */
    unsigned short    m_wVDOPtimes10;    /* VDOP (0.1 units) */
    unsigned short    m_wWAASMask;       /* Bits 0-1: tracked sats, Bits 2-3:
                                           used sats, Bits 5-9 WAAS PRN 1 minus
                                           120, Bits 10-14 WAAS PRN 1 minus 120 */
    unsigned short    m_wChecksum;       /* sum of all bytes of the header and data */
    unsigned short    m_wCRLF;           /* Carriage Return Line Feed */
} SBinaryMsg2;                          /* length = 8 + 16 + 2 + 2 = 28 */
```

**Additional
Information**

Message has a BlockID of 2 and is 16 bytes, excluding the header and epilogue

**Related
Commands**

[JBIN](#)

Bin62 Message

Message Type [Binary](#)

Description GLONASS almanac information

Command Format to Request Message \$JBIN, 62, R<CR><LF>

where:

- '62' = Bin62 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
SV	Satellite to which this data belongs	Byte	1	
Ktag_ch	Proprietary data	Byte	1	
Spare1	Spare, keeps alignment to 4 bytes	Unsigned short	2	
Strings[3]	GLONASS almanac data (36 bytes) <ul style="list-style-type: none"> • 0 & 1 = Two almanac SFs • 3= SF 5 	SGLONASS string	36	

Structure

```
typedef struct
{
    SUnionMsgHeader  m_sHead;
    unsigned char    m_bySV;          /* The satellite to which this data belongs. */
    unsigned char    m_byKtag_ch;     /* Proprietary data */
    unsigned short   m_wSpare1;       /* Spare, keeps alignment to 4 bytes */
    SGLONASS_String  m_asStrings[3]; /* glonass almanac data (36 bytes)
                                     0 & 1 = Two almanac SFs, 3= SF 5*/
    unsigned short   m_wChecksum;     /* sum of all bytes of the header and data */
    unsigned short   m_wCRLF;         /* Carriage Return Line Feed */
} SBinaryMsg62; /* length = 8 + (40) + 2 + 2 = 52 */
```

Additional Information

Related Commands [JBIN](#)

Bin65 Message

Message Type [Binary](#)

Description GLONASS ephemeris information

Command Format to Request Message \$JBIN, 65, R<CR><LF>

where:

- '65' = Bin65 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
SV	Satellite to which this data belongs	Byte	1	
Ktag	Satellite K Number + 8	Byte	1	
Spare1	Spare, keeps alignment to 4 bytes	Unsigned short	2	
TimeReceivedInSeconds	Time at which this arrived	Unsigned long	4	
Strings[5]	First five strings of GLONASS frame (60 bytes)	SGLONASS string	60	

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char   m_bySV;           /* The satellite to which this data belongs. */
    unsigned char   m_byKtag;         /* The satellite K Number + 8. */
    unsigned short  m_wSpare1;        /* Spare, keeps alignment to 4 bytes */
    unsigned long   m_ulTimeReceivedInSeconds; /* time at which this arrived */
    SGLONASS_String m_asStrings[5]; /* first 5 Strings of Glonass Frame (60 bytes) */
    unsigned short  m_wChecksum;      /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg65;                      /* length = 8 + (68) + 2 + 2 = 80 */
```

Additional Information

Related Commands [JBIN](#)

Bin66 Message

Message Type [Binary](#)

Description GLONASS L1 code and carrier phase information

Command Format to Request Message \$JBIN,66,R<CR><LF>

where:

- '66' = Bin66 message
- 'R' = message rate in Hz (20, 10, 2, 1, or 0)

Message Format

Component	Description	Type	Bytes	Values
Tow	Time in seconds	Double		
Week	GPS week number	Unsigned short		
Spare1	Spare 1 (zero)	Unsigned short		
Spare2	Spare 2 (zero)	Unsigned long		
L1Obs[CHANNELS_12]	12 sets of L1 (GLONASS) observations	SObsPacket		
L2Obs[CHANNELS_12]	12 sets of L2 (GLONASS) observations	SObsPacket		
L1CodeMSBsSlot[CHANNELS_12]	<i>See following</i>	Unsigned long		
<ul style="list-style-type: none"> • Bits 0-7 (8 bits) Satellite slot, 0 if no satellite • Bits 8-12 (5 bits) Spare bit • Bits 13- 31 (19 bits) Upper 19 bits of L1, LSB = 256 meters, MSB = 67108864 meters 				

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    double          m_dTow;           /* Time in seconds */
    unsigned short  m_wWeek;          /* GPS Week Number */
    unsigned short  m_wSpare1;        /* spare 1 (zero)*/
    unsigned long   m_ulSpare2;       /* spare 2 (zero)*/
    SObsPacket      m_asL1Obs[CHANNELS_12]; /* 12 sets of L1(Glonass)
                                           observations */
    SObsPacket      m_asL2Obs[CHANNELS_12]; /* 12 sets of L2(Glonass)
                                           observations */
    unsigned long   m_aulL1CodeMSBsSlot[CHANNELS_12]; /* array of 12 words.
                                                         bit 7:0 (8 bits) =
                                                         satellite Slot, 0 if no
                                                         satellite
                                                         bit 12:8 (5 bits) = spare
                                                         bit 31:13 (19 bits) =
                                                         upper 19 bits of L1
```



```

                                LSB = 256 meters
                                MSB = 67108864 meters */
unsigned short  m_wChecksum;    /* sum of all bytes of the header and data */
unsigned short  m_wCRLF;       /* Carriage Return Line Feed */
} SBinaryMsg66;               /* length = 8 + (352) + 2 + 2 = 364 */
```

Additional Information

**Related
Commands** [JBIN](#)

Bin69 Message

Message Type [Binary](#)

Description GLONASS L1 diagnostic information

Command Format to Request Message \$JBIN,69,R<CR><LF>

where:

- '69' = Bin69 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
SecOfWeek	Tow	Long		
L1usedNavMask	Mask of L1 channels used in nav solution	Unsigned short		
L2usedNavMask	Mask of L2 channels used in nav solution	Unsigned short		
ChannelData[CHANNELS_12]	Channel data 12X24 = 288	SGLONASSChan Data		
Week	Week	Unsigned short		
Spare01	Spare 1	Unsigned char		
Spare02	Spare 2	Unsigned char		

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    long            m_lSecOfWeek;      /* tow */
    unsigned short  m_wL1usedNavMask; /* mask of L1 channels used in nav solution */
    unsigned short  m_wL2usedNavMask; /* mask of L2 channels used in nav solution */
    SGLONASSChanData m_asChannelData[CHANNELS_12]; /* channel data 12X24 = 288 */
    unsigned short  m_wWeek;           /* week */
    unsigned char    m_bySpare01;      /* spare 1 */
    unsigned char    m_bySpare02;      /* spare 2 */
    unsigned short  m_wChecksum;       /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;           /* Carriage Return Line Feed */
} SBinaryMsg69; /* length = 8 + 300 + 2 + 2 = 312 */
```

Additional Information

Related Commands [JBIN](#)

Bin76 Message

Message Type [Binary](#)

Description GPS L1/L2 code and carrier phase information

Note: "Code" means pseudorange derived from code phase. "Phase" means range derived from carrier phase. This will contain cycle ambiguities.

Only the lower 16 bits of L1P code, L2P code and the lower 23 bits of carrier phase are provided. The upper 19 bits of the L1CA code are found in `m_aulCACodeMSBsPRN[]`. The upper 19 bits of L1P or L2P must be derived using the fact L1P and L2P are within 128 m (419.9 ft) of L1CA.

To determine L1P or L2P:

1. Use the lower 16 bits provided in the message.
2. Set the upper bits to that of L1CA.
3. Add or subtract on LSB of the upper bits (256 meters (839.9 feet)) so that L1P or L2P are within 1/2 LSB (128 m (419.9 ft))

The carrier phase is in units of cycles, rather than meters, and is held to within 1023 cycles of the respective code range. Only the lower $16+7 = 23$ bits of carrier phase are transmitted in Bin 76.

To determine the remaining bits:

1. Convert the respective code range (determined above) into cycles by dividing by the carrier wavelength. This is the nominal reference phase.
2. Extract the 16 and 7 bit blocks of carrier phase from bin 76 and arrange it to form the lower 23 bits of carrier phase.
3. Set the upper bits (bit 23 and above) equal to those of the nominal reference phase
4. Add or subtract the least significant upper bit (8192 cycles) so that carrier phase most closely agrees with the nominal reference phase (to within 4096 cycles).

Command Format to Request Message \$JBIN, 76, R<CR><LF>

where:

- '76' = Bin76 message
- 'R' = message rate in Hz (20, 10, 2, 1, 0, or .2)

Message Format

Component	Description	Type	Bytes	Values
TOW	Predicted GPS time in seconds	Double	8	
Week	GPS week number	Unsigned short	2	
Spare1		Unsigned long	2	
Spare2		Unsigned long	4	
<i>L2PSatObs[12]</i> (array for next 3	<i>L2 satellite observation data</i>	<i>Structure array</i>	<i>12 x 12 =</i>	

<i>fields)</i>			144	
CS_TT_W3_SNR	See following	Unsigned long	4	
<ul style="list-style-type: none"> Bits 0-11 (12 bits) SNR; $10.0 \times \log_{10}(0.1164 \times \text{SNR_value})$ Bits 12-14 (3 bits) Cycle Slip Warn (warning for potential 1/2 cycle slips); a warning exists if any of these bits are set Bit 15 (1 bit) Long Track Time; 1 if Track Time > 25.5 sec (0 otherwise) Bits 16-23 (8 bits) Track Time (signal tracking time in seconds); LSB = 0.1 seconds; Range = 0 to 25.5 seconds Bits 24-31 (8 bits) Cycle Slips; increments by 1 every cycle slip with natural roll-over after 255 				
P7_Doppler_FL	See following	Unsigned long	4	
<ul style="list-style-type: none"> Bit 0 (1 bit) Phase Valid (Boolean); 1 if valid phase (0 otherwise) Bits 1-23 (23 bits) Doppler (magnitude of Doppler); LSB = 1/512 cycle/sec; Range = 0 to 16384 cycle/sec Bit 24 (1 bit) Doppler Sign (sign of Doppler); 1 = negative, 0 = positive Bits 25-31 (7 bits) Carrier Phase (High port) (Upper 7 bits of the 23 bit carrier phase); LSB = 64 cycles, MSB = 4096 cycles 				
CideAndPhase	See following	Unsigned long	4	
<ul style="list-style-type: none"> Bits 0-15 (16 bits) Pseudorange (lower 16 bits of code pseudorange); LSB = 1/256 meters, MSB = 128 meters Note: For CA code, the upper 19 bits are given in L1CACodeMSBsPRN below Bits 16-31 (16 bits) Carrier Phase (lower 16 bits of the carrier phase); LSB = 1/1024 cycles, MSB = 32 cycles Note: The 7 MSBs are given in P7_Doppler_FL (see preceding row in this table) 				
L1CASatObs[15] (array for next 3 fields)	L1 satellite code observation data	Structure array	15 x 12 = 180	
CS_TT_W3_SNR	See following	Unsigned long	4	
<ul style="list-style-type: none"> Bits 0-11 (12 bits) SNR; $10.0 \times \log_{10}(0.1024 \times \text{SNR_value})$ Bits 12-14 (3 bits) Cycle Slip Warn (warning for potential 1/2 cycle slips); a warning exists if any of these bits are set Bit 15 (1 bit) Long Track Time; 1 if Track Time > 25.5 sec (0 otherwise) Bits 16-23 (8 bits) Track Time (signal tracking time in seconds); LSB = 0.1 seconds; Range = 0 to 25.5 seconds Bits 24-31 (8 bits) Cycle Slips; increments by 1 every cycle slip with natural roll-over after 255 				
P7_Doppler_FL	See following	Unsigned long	4	
<ul style="list-style-type: none"> Bit 0 (1 bit) Phase Valid (Boolean); 1 if valid phase (0 otherwise) Bits 1-23 (23 bits) 				

Doppler (magnitude of Doppler); LSB = 1/512 cycle/sec; Range = 0 to 16384 cycle/sec <ul style="list-style-type: none"> • Bit 24 (1 bit) Doppler Sign (sign of Doppler); 1 = negative, 0 = positive • Bits 25-31 (7 bits) Carrier Phase (High port) (Upper 7 bits of the 23 bit carrier phase): LSB = 64 cycles, MSB = 4096 cycles 				
CideAndPhase	See following	Unsigned long	4	
<ul style="list-style-type: none"> • Bits 0-15 (16 bits) Pseudorange (lower 16 bits of code pseudorange); LSB = 1/256 meters, MSB = 128 meters Note: For CA code, the upper 19 bits are given in L1CACodeMSBsPRN[] below • Bits 16-31 (16 bits) Carrier Phase (lower 16 bits of the carrier phase); LSB = 1/1024 cycles, MSB = 32 cycles Note: The 7 MSBs are given in P7 Doppler FL (see preceding row in this table) 				
L1CACodeMSBsPRN[15]	L1CA code observation	Array of 15 Unsigned long	15 x 4 = 60	See following
<ul style="list-style-type: none"> • Bits 0-7 (8 bits) PRN (space vehicle ID); PRN = 0 if no data • Bits 8-12 (5 bits) Unused • Bits 13-31 (19 bits) L1CA Range (upper 19 bits of L1CA); LSB = 256 meters, MSB = 67,108,864 meters 				
L1PCode[12]	L1(P) code observation data	Array of 12 Unsigned long	12 x 4 = 48	See following
<ul style="list-style-type: none"> • Bits 0-15 (16 bits) L1P Range (lower 16 bits of the L1P code pseudorange); LSB = 1/256 meters, MSB = 128 meters • Bits 16-27 (12 bits) L1P SNR (L1P signal-to-noise ratio); SNR = 10.0 x log(0.1164 x SNR_value), if 0, then L1P channel not tracked • Bits 28-31 (4 bits) Unused 				
wCeckSum	Sum of all bytes of header and data	Unsigned short	2	
wCRLF	Carriage return line feed	Unsigned short	2	

Structure

```

typedef struct
{
    SUnionMsgHeader m_sHead;
    double          m_dTow;           /* GPS Time in seconds */
    unsigned short  m_wWeek;          /* GPS Week Number */
    unsigned short  m_wSpare1;        /* spare 1 (zero) */
    unsigned long   m_ulSpare2;       /* spare 2 (zero) */
    SObsPacket      m_asL2PObs[CHANNELS_12]; /* 12 sets of L2(P) observations */
    SObsPacket      m_asL1CAObs[CHANNELS_L1_E]; /* 15 sets of L1(CA) observations */
    unsigned long   m_aulCACodeMSBsPRN[CHANNELS_L1_E]; /* array of 15 words.
                                                    bit 7:0 (8 bits) = satellite
                                                    PRN, 0 if no satellite
                                                    bit 12:8 (5 bits) = spare
                                                    bit 31:13 (19 bits) = upper
                                                    19 bits of L1CA
                                                    LSB = 256 meters
                                                    MSB = 67108864 meters */
    unsigned long   m_aul1Pword[CHANNELS_12]; /* array of 12 words relating to L1(P)
                                                    code. Bit 0-15 (16 bits) lower 16
                                                    bits of the L1P code pseudo range.

```

```
LSB = 1/256 meters
MSB = 128 meters
Bits 16-27 (12 bits) = L1P SNR_value
SNR = 10.0*log10( 0.1164*SNR_value)
If Bits 16-27 all zero, no L1P track
Bits 28-31 (4 bits) spare */
unsigned short m_wChecksum; /* sum of all bytes of the header and data */
unsigned short m_wCRLF; /* Carriage Return Line Feed */
} SBinaryMsg76; /* length = 8 + (448) + 2 + 2 = 460 */
```

Additional Information

**Related
Commands** [JBIN](#)

Bin80 Message

Message Type [Binary](#)

Description SBAS data frame information

Command Format to Request Message

\$JBIN, 80, R<CR><LF>

where:

- '80' = Bin80 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
PRN	Broadcast PRN	Unsigned short	2	
Spare	Not used at this time	Unsigned short	2	Future use
MsgSecOfWeek	Seconds of week for message	Unsigned long	4	
WaasMsg[8]	250-bit WAAS message (RTCA DO0229). 8 unsigned longs, with most significant bit received first.	Unsigned long	4 x 8 = 32	

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short m_wPRN;           /* Broadcast PRN */
    unsigned short m_wSpare;         /* spare (zero) */
    unsigned long m_ulMsgSecOfWeek;  /* Seconds of Week For Message */
    unsigned long m_aulWaasMsg[8];   /* Actual 250 bit waas message*/
    unsigned short m_wChecksum;      /* sum of all bytes of the headerand data */
    unsigned short m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg80;                    /* length = 8 + (40) + 2 + 2 = 52 */
```

Additional Information Message has a BlockID of 80 and is 40 bytes, excluding the header and epilogue

Related Commands [JBIN](#)

Bin89 Message

Message Type [Binary](#)

Description SBAS satellite tracking information (supports three SBAS satellites)

Command Format to Request Message \$JBIN,89,R<CR><LF>

where:

- '89' = Bin89 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
GPSSecOfWeek	GPS tow integer sec	Long		
MaskSBASTracked	SBAS satellites tracked, bit mapped 0..3	Byte		
MaskSBASUSED	SBAS satellites used, bit mapped 0..3	Byte		
Spare	Spare	Unsigned short		
ChannelData[CHANNELS_SBAS_E]	SBAS channel data	SChannelData		

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    long m_lGPSSecOfWeek; /* GPS tow integer sec */
    unsigned char m_byMaskSBASTracked; /* SBAS Sats Tracked, bit mapped 0..3 */
    unsigned char m_byMaskSBASUSED; /* SBAS Sats Used, bit mapped 0..3 */
    unsigned short m_wSpare; /* spare */
    SChannelData m_asChannelData[CHANNELS_SBAS_E]; /* SBAS channel data */
    unsigned short m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short m_wCRLF; /* Carriage Return Line Feed */
} SBinaryMsg89; /* length = 8 + 80 + 2 + 2 = 92 */
```

Additional Information

Related Commands [JBIN](#)

Bin93 Message

Message Type [Binary](#)

Description SBAS ephemeris information

Command Format to Request Message \$JBIN, 93, R<CR><LF>

where:

- '93' = Bin93 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
SV	Satellite to which this data belongs	Unsigned short	2	
Spare	Not used at this time	Unsigned short	2	Future use
TOWSecOfWeek	Time at which this arrived (LSB = 1 sec)	Unsigned long	4	
IODE		Unsigned short	2	
URA	Consult the ICD-GPS-200 for definition in Appendix A	Unsigned short	2	
TO	Bit 0 = 1 sec	Long	4	
XG	Bit 0 = 0.08 m	Long	4	
YG	Bit 0 = 0.08 m	Long	4	
ZG	Bit 0 = 0.4 m	Long	4	
XGDot	Bit 0 = 0.000625 m/sec	Long	4	
YXDot	Bit 0 = 0.000625 m/sec	Long	4	
ZGDot	Bit 0 = 0.004 m/sec	Long	4	
XGDotDot	Bit 0 = 0.0000125 m/sec/sec	Long	4	
YGDotDot	Bit 0 = 0.0000125 m/sec/sec	Long	4	
ZGDotDot	Bit 0 = 0.0000625 m/sec/sec	Long	4	
Gf0	Bit 0 = 2 ⁻³¹ sec	Unsigned short	2	
Gf0Dot	Bit 0 = 2 ⁻⁴⁰ sec/sec	Unsigned short	2	

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short m_wSV; /* The satellite to which this data belongs. */
}
```

```
unsigned short m_wWeek; /* Week corresponding to m_lTOW*/
unsigned long m_lSecOfWeekArrived; /* time at which this arrived (LSB = 1sec) */
unsigned short m_wIODE;
unsigned short m_wURA; /* See 2.5.3 of Global Pos Sys Std Pos Service Spec */
long m_lTOW; /* Sec of WEEK Bit 0 = 1 sec */
long m_lXG; /* Bit 0 = 0.08 m */
long m_lYG; /* Bit 0 = 0.08 m */
long m_lZG; /* Bit 0 = 0.4 m */
long m_lXGDot; /* Bit 0 = 0.000625 m/sec */
long m_lYGDot; /* Bit 0 = 0.000625 m/sec */
long m_lZGDot; /* Bit 0 = 0.004 m/sec */
long m_lXGDotDot; /* Bit 0 = 0.0000125 m/sec/sec */
long m_lYGDotDot; /* Bit 0 = 0.0000125 m/sec/sec */
long m_lZGDotDot; /* Bit 0 = 0.0000625 m/sec/sec */
short m_nGf0; /* Bit 0 = 2**-31 sec */
short m_nGf0Dot; /* Bit 0 = 2**-40 sec/sec */
unsigned short m_wChecksum; /* sum of all bytes of the header and data */
unsigned short m_wCRLF; /* Carriage Return Line Feed */
} SBinaryMsg93; /* length = 8 + (56) + 2 + 2 = 68 */
```

**Additional
Information**

Message has a BlockID of 93 and is 45 bytes, excluding the header and epilogue

**Related
Commands**

[JBIN](#)

Bin94 Message

Message Type [Binary](#)

Description Ionospheric and UTC conversion parameters

Command Format to Request Message \$JBIN, 94, R<CR><LF>

where:

- '94' = Bin94 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
a0, a1,a2, a3	AFCRL alpha parameters	Double	8 x 4 = 32	
b0, b1,b2, b3	AFCRL beta parameters	Double	8 x 4 = 32	
A0, A1	Coefficients for determining UTC time	Double	8 x 2 = 16	
tot	Reference time for A0 and A1, second of GPS week	Unsigned long	4	
wnt	Current UTC reference week	Unsigned short	2	
wnlsf	Week number when dtlsf becomes effective	Unsigned short	2	
dn	Day of week (1-7) when dtlsf becomes effective	Unsigned short	2	
dtls	Cumulative past leap	Short	2	
dtlsf	Scheduled future leap	Short	2	
Spare	Not used at this time	Short	2	Future use

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    /* Iono parameters. */
    double m_a0,m_a1,m_a2,m_a3; /* AFCRL alpha parameters. */
    double m_b0,m_b1,m_b2,m_b3; /* AFCRL beta parameters. */
    /* UTC conversion parameters. */
    double m_A0,m_A1; /* Coeffs for determining UTC time. */
    unsigned long m_tot; /* Reference time for A0 & A1, sec of GPS week. */
    unsigned short m_wnt; /* Current UTC reference week number. */
    unsigned short m_wnlsf; /* Week number when dtlsf becomes effective. */
    unsigned short m_dn; /* Day of week (1-7) when dtlsf becomes effective. */
    short m_dtls; /* Cumulative past leap seconds. */
    short m_dtlsf; /* Scheduled future leap seconds. */
    unsigned short m_wSpare1; /* spare 4 (zero)*/
    unsigned short m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short m_wCRLF; /* Carriage Return Line Feed */
}
```

```
} SBinaryMsg94;          /* length = 8 + (96) + 2 + 2 = 108 */
```

**Additional
Information**

Message has a BlockID of 94 and is 96 bytes, excluding the header and epilogue

**Related
Commands**

[JBIN](#)

Bin95 Message

Message Type [Binary](#)

Description GPS ephemeris information

Command Format to Request Message \$JBIN, 95, R<CR><LF>

where:

- '95' = Bin95 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
SV	Satellite to which this data belongs	Unsigned short	2	
Spare1	Not used at this time	Unsigned short	2	Future use
SecOfWeek	Time at which this arrived (LSB = 6)	Unsigned long	4	
SF1words[10]	Unparsed SF 1 message	Unsigned long	4 x 10 = 40	
SF2words[10]	Unparsed SF 2 message	Unsigned long	4 x 10 = 40	
SF3words[10]	Unparsed SF 3 message	Unsigned long	4 x 10 = 40	

Structure

```
typedef struct
{
    SUnionMsgHeader  m_sHead;
    unsigned short   m_wSV;           /* The satellite to which this data belongs. */
    unsigned short   m_wSpare1;       /* spare 1 (chan number (as zero 9/1/2004)*/
    unsigned long    m_TOW6SecOfWeek; /* time at which this arrived (LSB = 6sec) */
    unsigned long    m_SF1words[10];  /* Unparsed SF 1 message words. */
    unsigned long    m_SF2words[10];  /* Unparsed SF 2 message words. */
    unsigned long    m_SF3words[10];  /* Unparsed SF 3 message words. */
    /* Each of the subframe words contains
       one 30-bit GPS word in the lower
       30 bits, The upper two bits are ignored
       Bits are placed in the words from left to
       right as they are received */
    unsigned short   m_wChecksum;     /* sum of all bytes of the header and data */
    unsigned short   m_wCRLF;         /* Carriage Return Line Feed */
} SBinaryMsg95;                     /* length = 8 + (128) + 2 + 2 = 140 */
```

Additional Information Message has a BlockID of 95 and is 128 bytes, excluding the header and epilogue

Related Commands [JBIN](#)

Bin96 Message

Message Type [Binary](#)

Description GPS L1 code and carrier phase information

Command Format to Request Message \$JBIN,96,R<CR><LF>

where:

- '96' = Bin96 message
- 'R' = message rate in Hz (20, 10, 2, 1, or 0)

Message Format

Component	Description	Type	Bytes	Values
Spare1	Not used at this time	Unsigned short	2	Future use
Week	GPS week number	Unsigned short	2	
TOW	Predicted GPS time in seconds	Double	8	
UNICS_TT_SNR_PRN[12]	<i>See following</i>	Unsigned long	4	
<ul style="list-style-type: none"> • Bits 0-7 (8 bits) Pseudorandom noise; PRN is 0 if no data • Bits 8-15 (8 bits) Signal-to noise ratio (SNR); $SNR = 10.0 * \log_{10} (0.8192 * SNR)$ • Bits 16-23 (8 bits) PhaseTrackTime (PTT); in units of 1/10 sec; range=0 to 25 sec (if greater than 25 see UIDoppler_FL[12] below) • Bits 24-31 (8 bits) CycleSlip Counter (CSC); increments by 1 every cycle with natural rollover after 255 				
UIDoppler_FL[12]	<i>See following</i>	Unsigned long	4	
<ul style="list-style-type: none"> • Bit 0 (1 bit) Phase; Location 0; 1 if valid (0 otherwise) • Bit 1 (1 bit) TrackTime; 1 if track time > 25.5 seconds (0 otherwise) • Bits 2-3 (2 bits) Unused • Bits 4-31 (28 bits) Doppler; Signed (two's complement) Doppler in units of m/sec x 4096. (i.e., $LSB = 1/4096$), range = +/- 32768 m/sec. Computed as phase change over 1/10 sec. 				
PseudoRange[12]	Pseudorange	Double	8	
Phase[12]	Phase (m) L1 wave = 0.190293672798365	Double	8	

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned short  m_wSpare1;      /* spare 1 (zero) */
    unsigned short  m_wWeek;        /* GPS Week Number */
    double          m_dTow;         /* Predicted GPS Time in seconds */
    SObservations   m_asObvs[CHANNELS_12]; /* 12 sets of observations */
    unsigned short  m_wChecksum;    /* sum of all bytes of the header and data */
    unsigned short  m_wCRLF;        /* Carriage Return Line Feed */
} SBinaryMsg96;                   /* length = 8 + (300) + 2 + 2 = 312 */
```

Additional Information Message has a BlockID of 96 and is 300 bytes, excluding the header and epilogue

Related Commands [JBIN](#)

Bin97 Message

Message Type [Binary](#)

Description Processor statistics

Command Format to Request Message \$JBIN, 97, R<CR><LF>

where:

- '97' = Bin97 message
- 'R' = message rate in Hz (20, 10, 2, 1, 0, or .2)

Message Format

Component	Description	Type	Bytes	Values
CPUFactor	CPU utilization factor Multiply by 450e-06 to get percentage of spare CPU that is available Note: This field is only relevant on the old SLX platforms and Eclipse platform. It is not relevant for the Crescent receivers.	Unsigned long	4	Positive
MissedSubFrame	Total number of missed sub frames in the navigation message since power on	Unsigned short	2	Positive
MaxSubFramePnd	Max sub frames queued for processing at any one time	Unsigned short	2	Positive
MissedAccum	Total number of missed code accumulation measurements in the channel tracking loop	Unsigned short	2	Positive
MissedMeas	Total number missed pseudorange measurements	Unsigned short	2	Positive
Spare 1	Not used at this time	Unsigned long	4	Future use
Spare 2	Not used at this time	Unsigned long	4	Future use
Spare 3	Not used at this time	Unsigned long	4	Future use
Spare 4	Not used at this time	Unsigned short	2	Future use
Spare 5	Not used at this time	Unsigned short	2	Future use

Structure	<pre>typedef struct { SUnionMsgHeader m_sHead; unsigned long m_ulCPUFactor; /* CPU utilization Factor (%=multby 450e-6) */ unsigned short m_wMissedSubFrame; /* missed subframes */ unsigned short m_wMaxSubFramePend; /* max subframe pending */ unsigned short m_wMissedAccum; /* missed accumulations */ unsigned short m_wMissedMeas; /* missed measurements */ unsigned long m_ulSpare1; /* spare 1 (zero) */ unsigned long m_ulSpare2; /* spare 2 (zero) */ unsigned long m_ulSpare3; /* spare 3 (zero) */ unsigned short m_wSpare4; /* spare 4 (zero) */ unsigned short m_wSpare5; /* spare 5 (zero) */ unsigned short m_wChecksum; /* sum of all bytes of the header and data */ unsigned short m_wCRLF; /* Carriage Return Line Feed */ } SBinaryMsg97; /* length = 8 + (28) + 2 + 2 = 40 */</pre>
------------------	---

Additional Information	Message has a BlockID of 97 and is 28 bytes, excluding the header and epilogue
-------------------------------	--

Related Commands	JBIN
-------------------------	----------------------

Bin98 Message

Message Type [Binary](#)

Description Satellite and almanac information

Command Format to Request Message \$JBIN, 98, R<CR><LF>

where:

- '98' = Bin98 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
AlmanData[8]	SV data, 8 at a time	SSVAlmanData		See following
LastAlman	Last almanac processed	Byte	1	0 to 31
IonoUTCVFlag	Flag that is set when ionosphere modeling data is extracted from the GPS sub frame 4	Byte	1	0 = not logged 2 = valid
Spare	Not used at this time	Unsigned short	2	Future use

Structure

```
typedef struct
{
    SUnionMsgHeader  m_sHead;
    SSVAlmanData     m_asAlmanData[8]; /* SV data, 8 at a time */
    unsigned char    m_byLastAlman;    /* last almanac processed */
    unsigned char    m_byIonoUTCVFlag; /* iono UTC flag */
    unsigned short   m_wSpare;         /* spare */
    unsigned short   m_wChecksum;      /* sum of all bytes of the header and data */
    unsigned short   m_wCRLF;          /* Carriage Return Line Feed */
} SBinaryMsg98; /* length = 8 + (64+1+1+2) + 2 + 2 = 80 */
```

Additional Information Message has a BlockID of 98 and is 68 bytes, excluding the header and epilogue

Related Commands [JBIN](#)

Bin99 Message

Message Type [Binary](#)

Description GPS L1 diagnostic information

Command Format to Request Message \$JBIN, 99, R<CR><LF>

where:

- '99' = Bin99 message
- 'R' = message rate in Hz (1 or 0)

Message Format

Component	Description	Type	Bytes	Values
NavMode	Navigation mode data (lower 3 bits hold the GPS mode, upper bit set if differential is available)	Byte	1	Lower 3 bits take on the values: 0 = time not valid 1 = No fix 2 = 2D fix 3 = 3D fix Upper bit (bit 7) is 1 if differential is available
UTCTimeDiff	Whole seconds between UTC and GPS time (GPS minus UTC)	Byte	1	Positive
GPSWeek	GPS week associated with this message	Unsigned short	2	0 to 65536
GPSTimeofWeek	GPS tow (sec) associated with this message	Double	8	0.0 to 604800.0
sChannelData[CHANNELS_12]	Channel data	SChannelData	12 x 24 = 288	
ClockErrAtL1	Clock error of the GPS clock oscillator at L1 frequency in Hz	Short	2	-32768 to 32768
Spare	Not used at this time	Unsigned short	2	Future use

Structure

```
typedef struct
{
    SUnionMsgHeader m_sHead;
    unsigned char m_byNavMode; /* Nav Mode FIX_NO, FIX_2D, FIX_3D
                               (high bit =has_diff) */
    char m_cUTCTimeDiff; /* whole Seconds between UTC and GPS */
    unsigned short m_wGPSWeek; /* GPS week */
    double m_dGPSTimeOfWeek; /* GPS tow */
    SChannelData m_asChannelData[CHANNELS_12]; /* channel data */
    short m_nClockErrAtL1; /* clock error at L1, Hz */
    unsigned short m_wSpare; /* spare */
    unsigned short m_wChecksum; /* sum of all bytes of the header and data */
    unsigned short m_wCRLF; /* Carriage Return Line Feed */
} SBinaryMsg99; /* length = 8 + 304 + 2 + 2 = 316 */
```

Additional Information Message has a BlockID of 99 and is 304 bytes, excluding the header and epilogue

Related Commands [JBIN](#)

CRMSK Message

Message Type [Beacon](#)

Description Operational status message of SBX

Command Format to Request Message \$GPCRQ,MSK<CR><LF>

Message Format \$CRMSK,fff.f,X,ddd,Y,n*CC<CR><LF>

where:

Response Component	Description
fff.f	Frequency, in kHz (283.5 to 325)
X	Tune mode (M = manual, A = automatic)
ddd	MSK bit rate, in bps (100 or 200)
Y	MSK rate selection mode (M = manual, A = automatic)
n	Period of output of performance status message, in seconds (0 to 100); see CRMSS
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [GPCRQ,MSK](#)

CRMSS Message

Message Type [Beacon](#)

Description Performance status message of SBX

Command Format to Request Message \$GPCRQ,MSS<CR><LF>

Message Format \$CRMSS,xx,yy,fff.f,ddd*CC<CR><LF>

where:

Response Component	Description
xx	Signal strength, in dB μ V/m
yy	Signal-to-noise ratio, in dB
fff.f	Frequency, in kHz (283.5 to 325)
ddd	MSK bit rate in bps (100 or 200)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [GPCRQ.MSS](#)

GLMLA Message

Message Type [GLONASS](#)

Description GLONASS almanac data

Contains complete almanac data for one GLONASS satellite. Multiple sentences may be transmitted, one for each satellite in the GLONASS constellation.

Command Format to Request Message

\$JASC, GLMLA, R[, OTHER] <CR><LF>

where

- R = message rate (in Hz) of (1 or 0)
- ,OTHER = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format

\$JASC, GLMLA, a.a, b.b, cc, d.d, ee, ffff, gg, hhhh, iiii, jjjjjj, kkkkkk, mmmmmm, nnnnnn, ppp, qq*hh<CR><LF>

where:

Message Component	Description
a.a	Total number of sentences
b.b	Sentence number
cc	Satellite ID (satellite slot) number
d.d	Calendar day count within the four year period beginning with the previous leap year
ee	Generalized health of the satellite and carrier frequency number respectively
ffff	Eccentricity
gg	DOT, rate of change of the draconitic circling time
hhhh	Argument of perigee
iiii	16 MSB of system time scale correction
jjjjjj	Correction to the average value of the draconitic circling time
kkkkkk	Time of the ascension node, almanac reference time
mmmmmm	Greenwich longitude of the ascension node
nnnnnn	Correction to the average value of the inclination angle
ppp	LSB of system time scale correction
qqq	Course value of the time scale shift

Example

Additional Information	Similar to the GPS message GPALM
-------------------------------	--

Related Commands	JASC.GL
-------------------------	-------------------------

GNSSPositionData Message

Message Type [NMEA 2000 CAN](#)

Description Detailed GPS position information

The GNSSPositionData message (PGN 0x1F805/129029) has an update rate of 1 Hz and DLC of 43, 47, or 51, dependent on the NumberOfReferenceStations.

Command Format to Request Message Message is continuously output on A100 CAN port

Message Format The following table provides the start bit, length (bit), value type, factor, and offset for the GNSSPositionData message.

Field Name	Start bit	Length (Bit)	Byte Order	Value Type	Factor
SequenceID	0	8	Intel	Unsigned	1
PositionDate	8	16	Intel	Unsigned	1
PositionTime	24	32	Intel	Unsigned	0.0001
LatitudeLow	56	32	Intel	Unsigned	1.00E-16
LatitudeHigh	88	32	Intel	Signed	4.29E-07
LongitudeLow	120	32	Intel	Unsigned	1.00E-16
LongitudeHigh	152	32	Intel	Signed	4.29E-07
AltitudeLow	184	32	Intel	Unsigned	1.00E-6
AltitudeHigh	216	32	Intel	Signed	4294.97
TypeOfSystem	248	4	Intel	Unsigned	1
GNSSMethod	252	4	Intel	Unsigned	1
GNSSIntegrity	256	2	Intel	Unsigned	1
GNSS_Reserved1	258	6	Intel	Unsigned	1
NumberOfSVs	264	8	Intel	Unsigned	1
HDOP	272	16	Intel	Signed	0.01
PDOP	288	16	Intel	Signed	0.01
GeodalSeparation	304	32	Intel	Signed	0.01
NumberOfReferenceStations	336	8	Intel	Unsigned	1
ReferenceStationType1	344	4	Intel	Unsigned	1
ReferenceStationID1	348	12	Intel	Unsigned	1
AgeOfDGNSSCorrections1	360	16	Intel	Unsigned	0.01

ReferenceStationType2	376	4	Intel	Unsigned	1
ReferenceStationID2	380	12	Intel	Unsigned	1
AgeOfDGNSSCorrections2	392	16	Intel	Unsigned	0.01

The following table provides the offset, minimum and maximum values, unit, and comment for the GNSSPositionData message.

Field Name	Offset	Min	Max	Unit	Comment
SequenceID	0	0	255		An upward counting number used to tie related information together between different PGNS
PositionDate	0	0	65532	day	Days since January 1, 1970. Date is relative to UTC time.
PositionTime	0	0	86401	sec	24 hour clock, 0=midnight, time is in UTC
LatitudeLow	0	0	4.29E-07	deg	Latitude referenced to WGS-84
LatitudeHigh	0	-90	90	deg	Latitude referenced to WGS-84
LongitudeLow	0	0		deg	Longitude referenced to WGS-84
LongitudeHigh	0	-180		deg	Longitude referenced to WGS-84
AltitudeLow	0	0		m	Altitude referenced to WGS-84
AltitudeHigh	0	-9.22 E+12		m	Altitude referenced to WGS-84
TypeOfSystem	0	0	4		0x0 GPS 0x1 GLONASS 0x2 GPS and GLONASS 0x3 GPS and SBAS, (WAAS/EGNOS) 0x4 GPS and SBAS and GLONASS
GNSSMethod	0	0	15		0x0 No GPS 0x1 GNSS fix 0x2 DGNSS fix 0x3 Precise GNSS 0x4 RTK fixed integer 0x5 RTK float 0x6 Estimated (DR) mode 0x7 Manual input 0x8 Simulate mode 0xE Error

GNSSIntegrity	0	0	3		0x0 No integrity checking 0x1 Safe 0x2 Caution 0x3 Unsafe
GNSS_Reserved1	0	0	63		
NumberOfSVs	0	0	252		Numeric count, event counter
HDOP	0	- 327.64	327.64		Dilution of Precision (DOP) indicates the contribution of satellite configuration geometry to positioning error
PDOP	0	- 327.64	327.64		Dilution of Precision (DOP) indicates the contribution of satellite configuration geometry to positioning error
GeodalSeparation	0	-2.15 E+07	2.15 E+07	m	The difference between the earth ellipsoid and mean sea- level (period), defined by the reference datum used in the position solution. '-' indicates mean sea-level below ellipsoid
NumberOfReferenceStations	0	0	252		Number of reference stations reported
ReferenceStationType1	0	0	15		0x0 GPS 0x1 GLONASS 0xE Error
ReferenceStationID1	0	0	4095		Reference station ID
AgeOfDGNSSCorrections1	0	0	655.32	sec	Age of differential corrections
ReferenceStationType2	0	0	15		0x0 GPS 0x1 GLONASS 0xE Error
ReferenceStationID2	0	0	4095		Reference station ID
AgeOfDGNSSCorrections2	0	0	655.32	sec	Age of differential corrections

Additional Information

Related Commands

GNSSPositionRapidUpdates Message

Message Type [NMEA 2000 CAN](#)

Description Abbreviated GPS position information

The GNSSPositionRapidUpdates message (PGN 0x1F801/129025) has an update rate equal to the subscribed rate (default of 10 Hz) and DLC of 8.

Command Format to Request Message Message is continuously output on A100 CAN port

Message Format The following table provides the start bit, length (bit), value type, factor, and offset for fields of the GNSSPositionRapidUpdates message.

Field Name	Start bit	Length (Bit)	Byte Order	Value Type	Factor	Offset	Min	Max	Unit
Latitude	0	32	Intel	Signed	0.0000001	0	-90	90	deg
Longitude	32	32	Intel	Signed	0.0000001	0	-180	180	deg

Additional Information

Related Commands

GPALM Message

Message Type [Data](#)

Description Message number (individual and total), week number, satellite health, and the almanac data for each satellite in the GPS constellation, up to a maximum of 32 messages

Command Format to Request Message \$JASC,GPALM,R[,OTHER] <CR><LF>

where

- R = message rate (in Hz) of (1 or 0)
- ,OTHER = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPALM,A.B,C.D,E,F,hh,hhhh,...*CC<CR><LF>

where:

Response Component	Description
A	Total number of messages
B	Message number
C	Satellite PRN number
D	GPS week number (0-1023)
E	Satellite health (bits 17-24 of message)
F	Eccentricity
hh	t index OA, almanac reference time
hhhh	sigma index 1, inclination angle

Example

```
$>
$GPALM,31,1,02,1617,00,50F6,0F,FD98,FD39,A10CF3,81389B,423632,BD913C,148,001*3C
$GPALM,31,2,03,1617,00,71B9,0F,F6C2,FD45,A10C96,2B833C,131DB4,BA69EE,2B1,001*3A
$GPALM,31,3,04,1617,00,4F01,0F,FD03,FD39,A10BFC,1C6C35,42EDB1,35B537,112,003*45
$GPALM,31,4,05,1617,00,121B,0F,08C8,FD61,A10C5C,09CA99,6D7257,021B32,79F,7FE*3E
$GPALM,31,5,06,1617,00,337F,0F,FB6B,FD49,A10CC2,DBE103,161127,10CD11,18C,7FE*4A
.
.
.
$GPALM,31,29,30,1617,00,6A85,0F,0ADD,FD5C,A11A83,3F6243,EBCC46,E8548D,145,001*00
$GPALM,31,30,31,1617,00,4037,0F,1778,FD3E,A10C28,D62817,C32ADF,781125,01B,001*7E
$GPALM,31,31,32,1617,00,65B5,0F,0956,FD65,A10DD0,DD74BA,71125D,985AE3,751,7FE*72
```

Additional Information Similar to the GLONASS message [GLMLA](#)

Related Commands [JASC.GP](#)

GPDTM Message

Message Type [Data](#)

Description Datum reference

Command Format to Request Message \$JASC , GPDTM , R [, OTHER] <CR><LF>

where

- R = message rate (in Hz) of (1 or 0)
- ,OTHER = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPDTM , ccc , a , x . x , a , x . x , a , x . x , ccc * CC <CR><LF>

where:

Response Component	Description
ccc	Local datum (normally W84, but could be NAD83 when using beacon in North America)
a	Local datum subdivision code
x.x,a	Lat offset, in minutes, N/S
x.x,a	Lon offset, minutes, E/W
x.x	Altitude offset, in meters
ccc	Reference datum (always W84)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Example \$GPDTM , W84 , , 0 . 0 , N , 0 . 0 , E , 0 . 0 , W84 * CC <CR><LF>

Additional Information

Related Commands [JASC.GP](#)

GPGGA Message

Note: This topic provides information pertaining to GPS. The format is the same for the messages pertaining to GNSS and GLONASS (see [Additional Information](#) below).

Message Type [Data](#)

Description Detailed GPS position information (most frequently used NMEA 0183 data message)

Command Format to Request Message \$JASC , GPGGA , R[, OTHER] <CR><LF>

where

- R = message rate (in Hz) of 20, 10, 5, 4, 2, 1, 0, or .2 (0 turns off the message)
- ,OTHER = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGGA , HHMMSS . SS , DDMM . MMMMM , S , DDDMM . MMMMM , S , N , QQ , PP . P , AAAA . AA , M , \pm XX . XX , M , SSS , AAAA *CC<CR><LF>

where:

Message Component	Description
HHMMSS.SS	UTC time in hours, minutes, and seconds of the GPS position
DDMM.MMMMM	Latitude in degrees, minutes, and decimal minutes
S	S = N (North latitude) or S (South latitude)
DDDMM.MMMMM	Longitude in degrees, minutes, and decimal minutes
S	S = E (East longitude) or W (West longitude)
N	Quality indicator <ul style="list-style-type: none"> • 0 = no position • 1 = undifferentially corrected position (autonomous) • 2 = differentially corrected position (SBAS, DGPS, OmniSTAR VBS, L-Dif and e-Dif) • 4 = RTK fixed integer (Crescent RTK, Eclipse RTK), OmniSTAR XP/HP converged • 5 = OmniSTAR XP/HP converging
QQ	Number of satellites used in position computation
PP.P	Horizontal dilution of precision (HDOP), ranging from 0.0 to 9.9
AAAA.AA	Antenna altitude
M	Altitude units, in meters
+/-XX.XX	Geoidal separation (needs geoidal height option)

M	Geoidal separation units, in meters
SSS	Age of differential corrections, in seconds
AAAA	Reference station identification
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Example \$GPGGA,175250.00,3333.42646711,N,11153.35317335,W,2,07,1.3,406.854,M,-26.294,M,10.4,0100*4F

Additional Information This message provides information specific to the satellite system identified by the first two characters of the message.

GP GGA - GPS information
GN GGA - GNSS information
GL GGA - GLONASS information

The [JNMEA.GGAALLGNSS](#) command significantly affects the output of the GGA message. If you are tracking more than GPS signals, Hemisphere GPS highly recommends that you review this command.

Related Commands [JASC.GP](#), [JASC.GN](#), [JASC.GL](#), [JNMEA.GGAALLGNSS](#)

GPGLL Message

Note: This topic provides information pertaining to GPS. The format is the same for the messages pertaining to GNSS and GLONASS (see [Additional Information](#) below).

Message Type [Data](#)

Description Latitude and longitude data

Command Format to Request Message \$JASC,GPGLL,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGLL,DDMM.MMMMM,S,DDDMM.MMMMM,S,HHMMSS.SS,S*CC<CR><LF>

where:

Message Component	Description
DDMM.MMMMM	Latitude in degrees, minutes, and decimal minutes
S	S = N (North latitude) or S (South latitude)
DDDMM.MMMMM	Longitude in degrees, minutes, and decimal minutes
S	S = E (East longitude) or W (West longitude)
HHMMSS.SS	UTC time in hours, minutes, and seconds of GPS position
S	Status, S = A (valid) or V (invalid)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information This message provides information specific to the satellite system identified by the first two characters of the message.

GPGLL - GPS information
 GNGLL - GNSS information
 GLGLL - GLONASS information

The [JNMEA,GGAALLGNSS](#) command significantly affects the output of the GLL message. If you are tracking more than GPS signals, Hemisphere GPS highly recommends that you review this command.

Related Commands [JASC,GP](#), [JASC,GN](#), [JASC,GL](#), [JNMEA,GGAALLGNSS](#)

GPGNS Message

Note: This topic provides information pertaining to GPS. The format is the same for the messages pertaining to GNSS and GLONASS (see [Additional Information](#) below).

Message Type [Data](#)

Description Fixes data for single or combined (GPS, GLONASS, possible future satellite systems, and systems combining these) satellite navigation systems

Command Format to Request Message \$JASC,GPGNS,R[,OTHER]<CR><LF>

where

- 'R' = message rate (in Hz) of 20, 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGNS,hhmmss.ss,llll.ll,a,yyyy.yy,a,mm,ss,h.h,a.a,g.g,d.d,r.r*CC<CR><LF>

where:

Message Component	Description
hhmmss.ss	UTC of position
llll.ll	Latitude, N/S
a	Latitude, N/S
yyyy.yy	Longitude, E/W
a	Longitude, E/W
mm	Mode indicator
ss	Total number of satellites in use, 00-99
h.h	HDOP
a.a	Antenna altitude, in meters, re: mean-sea-level (geoid)
g.g	Geoidal separation (in meters)
d.d	Age of differential data
r.r	Differential reference station ID
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional This message provides information specific to the satellite system identified by the first two characters of the

Information message.

GPGNS - GPS information

GNGNS - GNSS information

GLGNS - GLONASS information

The [JNMEA,GGAALLGNSS](#) command significantly affects the output of the GNS message. If you are tracking more than GPS signals, Hemisphere GPS highly recommends that you review this command.

**Related
Commands** [JASC,GP](#), [JASC,GN](#), [JASC,GL](#), [JNMEA,GGAALLGNSS](#)

GPGRS Message

Message Type [Data](#)

Description Supports Receiver Autonomous Integrity Monitoring (RAIM)

Command Format to Request Message \$JASC,GPGRS,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGRS.hhmmss.ss,m,x.x ... x.x*CC<CR><LF>

where:

Message Component	Description
hhmmss.ss	UTC time
m	Mode: 0 = residuals used to calculate the position given in the GPGBA or GPGNS message 1 = residuals were recomputed after the GPGBA or GPGNS message position was computed
x.x ... x.x	Range residuals, in meters, for satellites used in the navigation solution. Order must match order of satellite ID numbers in GPGSA message. When GPGRS message is used, the GPGSA and GPGSV messages are generally required with this message.
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC,GP](#)

GPGSA Message

Note: This topic provides information pertaining to GPS. The format is the same for the messages pertaining to GNSS and GLONASS (see [Additional Information](#) below).

Message Type [Data](#)

Description GPS DOP and active satellite information

Only satellites used in the position computation are present in this message. Null fields are present when data is unavailable due to the number of satellites tracked.

Command Format to Request Message \$JASC , GPGSA , R[, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGSA , A , B , CC . . . OO , P . P , Q . Q , R . R *CC <CR><LF>

where:

Message Component	Description
A	Satellite acquisition mode (M = manually forced to 2D or 3D, A = automatic swap between 2D and 3D)
B	Position mode (1 = fix not available, 2 = 2D fix, 3 = 3D fix)
CC to OO	Satellites used in the position solution, a null field occurs if a channel is unused
P.P	Position Dilution of Precision (PDOP) = 1.0 to 9.9
Q.Q	Horizontal Dilution of Precision (HDOP) 1.0 to 9.9
R.R	Vertical Dilution of Precision (VDOP) = 1.0 to 9.9
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information This message provides information specific to the satellite system identified by the first two characters of the message.

GPGSA - GPS information
 GNGSA - GNSS information
 GLGSA - GLONASS information

Related Commands [JASC.GP](#), [JASC.GN](#), [JASC.GL](#)

GPGST Message

Message Type [Data](#)

Description GNSS pseudorange error statistics and position accuracy

Command Format to Request Message \$JASC,GPGST,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGST,HHMMSS.SS,A.A,B.B,C.C,D.D,E.E,F.F,G.G*CC<CR><LF>

where:

Message Component	Description
HHMMSS.SS	UTC time in hours, minutes, and seconds of the GPS position
A.A	Root mean square (rms) value of the standard deviation of the range inputs to the navigation process. Range inputs include pseudoranges and differential GNSS (DGNSS) corrections.
B.B	Standard deviation of semi-major axis of error ellipse, in meters
C.C	Standard deviation of semi-minor axis of error ellipse, in meters
D.D	Error in Eclipse's semi major axis origination, in decimal degrees, true north
E.E	Standard deviation of latitude error, in meters
F.F	Standard deviation of longitude error, in meters
G.G	Standard deviation of altitude error, in meters
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

GPGSV Message

Note: This topic provides information pertaining to GPS. The format is the same for the message pertaining to GLONASS (see [Additional Information](#) below).

Message Type [Data](#)

Description GNSS satellite in view

Null fields occur where data is unavailable due to the number of satellites tracked.

Command Format to Request Message \$JASC,GPGSV,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPGSV,T,M,N,II,EE,AAA,SS,...II,EE,AAA,SS*CC<CR><LF>

where:

Message Component	Description
T	Total number of messages
M	Message number (1 to 3)
N	Total number of satellites in view
II	Satellite number
EE	Elevation, in degrees (0 to 90)
AAA	Azimuth (true), in degrees (0 to 359)
SS	Signal strength, in dB-Hz (0 - 99) To compare with SNR values found in Bin messages (such as Bin96) subtract 30 from this signal strength value for an approximate SNR value SS - 30 = SNR (from Bin message)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information This message provides information specific to the satellite system identified by the first two characters of the message.

GPGSV - GPS information

GLGSV - GLONASS information

If you request GNGSV the receiver will respond with GPGSV messages only.

Related Commands [JASC.GP](#), [JASC.GL](#)

GPHDG/HEHDG Message

Message Type [Data](#)

Description Magnetic deviation and variation for calculating magnetic or true heading

The message simulates data from a magnetic sensor although it does not actually contain one. The purpose of this message is to support older systems that may not be able to accept the HDT message that is recommended for use.

Command Format to Request Message \$JASC,GPHDG,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0 or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPHDG,s.s,d.d,D,v.v,V*CC<CR><LF>
or
\$HEHDG,s.s,d.d,D,v.v,V*CC<CR><LF>

where:

Message Component	Description
s.s	Magnetic sensor reading, in degrees
d.d	Magnetic deviation, in degrees
D	E = Easterly deviation, W = Westerly deviation
v.v	Magnetic variation, in degrees
V	E = Easterly deviation, W = Westerly deviation
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information You can change the HDG message header to either GP or HE using the [JATT,NMEAHE](#) command.

Related Commands [JASC,GP](#)

GPHDM/HEHDM Message

Message Type [Data](#)

Description Magnetic heading of the vessel derived from the true heading calculated

Command Format to Request Message \$JASC,GPHDM,R[,OTHER]<CR><LF>
 where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0 or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPHDM,x.x,M*CC<CR><LF>
 or
 \$HEHDM,x.x,M*CC<CR><LF>

where:

Message Component	Description
x.x	Current heading, in degrees
M	Indicates magnetic heading
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information You can change the HDM message header to either GP or HE using the [JATT.NMEAHE](#) command.

Related Commands [JASC.GP](#)

GPHDT/HEHDT Message

Message Type [Data](#)

Description True heading of the vessel

This is the direction that the vessel (antennas) is pointing and is not necessarily the direction of vessel motion (the course over ground).

Command Format to Request Message \$JASC,GPHDT,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0 or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPHDT,x.x,T*CC<CR><LF>
or
\$HEHDT,x.x,T*CC<CR><LF>

where:

Message Component	Description
x.x	Current heading, in degrees
T	Indicates true heading
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information You can change the HDT message header to either GP or HE using the [JATT.NMEAHE](#) command.

Related Commands [JASC.GP](#)

GPHEV Message

Message Type [Data](#)

Description Heave value in meters

Command Format to Request Message \$JASC , GPHEV , 1 <CR> <LF>

Message Format \$GPHEV , H , *CC <CR> <LF>

where:

Message Component	Description
H	Heave value, in meters
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

GPRMC Message

Message Type [Data](#)

Description Contains recommended minimum specific GNSS data

Command Format to Request Message \$JASC , GPRMC , R[, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPRMC , HHMMSS . SS , A , DDMM . MMM , N , DDDMM . MMM , W , Z . Z , Y . Y , DDMMYY , D . D , V
*CC<CR><LF>

where:

Message Component	Description
HHMMSS.SS	UTC time in hours, minutes, and seconds of the GPS position
A	Status (A = valid, V = invalid)
DDMM.MMM	Latitude in degrees, minutes, and decimal minutes
N	Latitude location (N = North latitude, S = South latitude)
DDDMM.MMM	Longitude in degrees, minutes, and decimal minutes
W	Longitude location (E = East longitude, W = West longitude)
Z.Z	Ground speed, in knots
Y.Y	Track made good, reference to true north
DDMMYY	UTC date of position fix in day, month, and year
D.D	Magnetic Variation, in degrees
V	Variation sense (E = East, W = West)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

GPROT/HEROT Message

Message Type [Data](#)

Description Vessel's rate of turn (ROT) information

Command Format to Request Message \$JASC , GPROT , R[, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0 or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPROT , x . x , A * CC <CR><LF>
or
\$HEROT , x . x , A * CC <CR><LF>

where:

Message Component	Description
x.x	Rate of turn in °/min (negative when the vessel bow turns to port)
A	Flag indicating the data is valid
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information You can change the ROT message header to either GP or HE using the [JATT,NMEAHE](#) command.

Related Commands [JASC.GP](#)

GPRRE Message

Message Type [Data](#)

Description Satellite range residuals and estimated position error

Command Format to Request Message \$JASC,GPRRE,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPRRE,N,II,RR...II,RR,HHH.H,VVV.V*CC<CR><LF>

where:

Message Component	Description
N	Number of satellites used in position computation
II	Satellite number
RR	Range residual, in meters
HHH.H	Horizontal position error estimate, in meters
VVV.V	Vertical position error estimate, in meters
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC_GP](#)

GPVTG Message

Message Type [Data](#)

Description Course over ground and ground speed

Command Format to Request Message \$JASC,GPVTG,R[,OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPVTG,TTT,T,MMM,M,NNN.NN,N,KKK.KK,K,X*CC<CR><LF>

where:

Message Component	Description
TTT	True course over ground (COG) in degrees (000 to 359)
T	True course over ground indicator (always 'T')
MMM	Magnetic course over ground in degrees (000 to 359)
M	Magnetic course over ground indicator (always 'M')
NNN.NN	Speed over ground in knots
N	Speed over ground in knots indicator (always 'N')
KKK.KK	Speed over ground in km/h
K	Speed over ground in km/h indicator (always 'K')
X	Mode A = Autonomous mode D = Differential mode E = Estimated (dead reckoning) mode M = Manual input mode S = Simulator mode N = Data not valid
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Example Sample message output:

\$GPVTG,103.85,T,92.79,M,0.14,N,0.25,K,D*1E

**Additional
Information**

**Related
Commands** [JASC.GP](#)

GPZDA Message

Message Type [Data](#)

Description UTC time and date information

Command Format to Request Message \$JASC , GPZDA , R [, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0, or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$GPZDA , HHMMSS . SS , DD , MM , YYYY , XX , YY *CC<CR><LF>

where:

Message Component	Description
HHMMSS.SS	UTC time in hours, minutes, and seconds of the GPS unit
DD	Day (0 to 31)
MM	Month (1 to 12)
YYYY	Year
XX	Local zone description in hours (-13 to 13)
YY	Local zone description in minutes (0 to 59)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

NMEACogSogData Message

Message Type [NMEA 2000 CAN](#)

Description GPS speed and direction information

The NMEACogSogData command (PGN 0x1F802/129026) has an update rate equal to the subscribed rate (default of 10 Hz) and DLC of 8.

Command Format to Request Message Message is continuously output on A100 CAN port

Message Format The following table describes the fields of the NMEACogSogData message:

Field Name	Start Bit	Length (Bit)	Byte Order	Value Type	Factor	Min	Max	Comment
NMEA_SequenceID	0	8	Intel	Unsigned	1	0	255	An upward counting number used to tie related information together between different PGNs
NMEA_Direction Reference	8	2	Intel	Unsigned	1	0	3	0x0 True north 0x1 Magnetic north 0x2 Error 0x3 Null
NMEA_Reserved1	10	6	Intel	Unsigned	1	0	63	
NMEA_Course OverGround	16	16	Intel	Unsigned	0.0001	0	6.5535	GPS based travel direction, in rad
NMEA_Speed OverGround	32	16	Intel	Unsigned	0.01	0	655.35	GPS based travel speed, in m/s
NMEA_Reserved2	48	16	Intel	Unsigned	1	0	65535	

Additional Information

Related Commands

PASHR Message

Message Type [Crescent Vector](#), [Data](#)

Description Time, heading, roll, and pitch data in one message

Command Format to Request Message

\$JASC , PASHR , R [, OTHER] <CR><LF>

where

- 'R' = message rate (0 = Off, 1 = On at 1Hz)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Message Format

\$PASHR , hhmmss.ss , HHH.HH , T , RRR.RR , PPP.PP , heave , rr.rrr , pp.ppp , hh.hhh , QF*CC <CR><LF>

where:

Message Component	Description
hhmmss.ss	UTC time
HHH.HH	Heading value in decimal degrees
T	True heading (T displayed if heading is relative to true north)
RRR.RR	Roll in decimal degrees (- sign will be displayed when applicable)
PPP.PP	Pitch in decimal degrees (- sign will be displayed when applicable)
heave	Heave, in meters
rr.rrr	Roll standard deviation in decimal degrees
pp.ppp	Pitch standard deviation in decimal degrees
hh.hhh	Heading standard deviation in decimal degrees
QF	Quality Flag <ul style="list-style-type: none"> • 0 = No position • 1 = All non-RTK fixed integer positions • 2 = RTK fixed integer position
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC,PASHR](#)

PSAT,GBS Message

Message Type [Data](#)

Description Used to support Receiver Autonomous Integrity Monitoring (RAIM)

Command Format to Request Message \$JASC,GPGBS,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$PSAT,GBS,hhmmss.ss,ll.l,LL.L,aa.a,ID,p.ppppp,b.b,s.s,flag*CC<CR><LF>

where:

Message Component	Description
hhmmss.ss	UTC time in hours, minutes, and seconds of the GGA or GNS fix associated with this sentence
ll.l	Expected error in latitude
LL.L	Expected error in longitude
aa.a	Expected error in altitude
ID	ID number of most likely failed satellite
p.ppppp	Probability of HPR fault
b.b	Estimate of range bias, in meters, on most likely failed satellite
s.s	Standard deviation of range bias estimate
flag	Based on horizontal radius: 0 = Good 1 = Warning 2 = Bad or Fault
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

PSAT,HPR Message

Message Type [Data](#)

Description Proprietary NMEA message that provides the heading, pitch, roll, and time in a single message

During normal operation heading and pitch are derived from GPS and roll comes from the inertial sensor. While coasting heading is based on gyro and pitch/roll are from the inertial sensor.

Command Format to Request Message \$JASC ,GPHPR ,R[, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 20, 10, 2, 1, 0 or .2 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$PSAT ,HPR ,time ,heading ,pitch ,roll ,type *CC <CR><LF>

where:

Message Component	Description
time	UTC time (HHMMSS.SS)
heading	Heading (degrees)
pitch	Pitch (degrees)
roll	Roll (degrees)
type	N for GPS derived heading G for gyro heading
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC.GP](#)

PSAT,INTLT Message

Message Type [Data](#)

Description Proprietary NMEA message that provides the tilt measurements from the internal inclinometers in degrees. It delivers an output of crude accelerometer measurements of pitch and roll with no temperature compensation or calibration for GPS heading/pitch/roll.

Pitch and roll are factory calibrated over temperature to be accurate to $\pm 3^{\circ}\text{C}$.

CAUTION: User calibration will clear out precise factory calibration.

Command Format to Request Message \$JASC,INTLT,R[,OTHER]<CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
 - ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)
-

Message Format \$PSAT,INTLT,pitch,roll*CC<CR><LF>

where:

Message Component	Description
pitch	Pitch (degrees)
roll	Roll (degrees)
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Additional Information

Related Commands [JASC,GP](#)

PSAT,RTKSTAT Message

Message Type [Data, Local Differential and RTK](#)

Description Contains the most relevant parameters affecting RTK

Command Format to Request Message \$JASC , PSAT , RTKSTAT , R[, OTHER] <CR><LF>

where

- 'R' = message rate in Hz of 1 or 0 (0 turns off the message)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets)

Message Format \$PSAT , RTKSTAT , MODE , TYP , AGE , SUBOPT , DIST , SYS , NUM , SNR , RSF , BSF , HAG*CC<CR><LF>

where:

Message Component	Description
MODE	FIX,FLT,DIF,AUT,NO
TYP	DFX,ROX,CMR,RTCM3,CMR+,...
AGE	Age of differential corrections
SUBOPT	Subscribed options
DIST	Distance to base in kilometers
SYS	Systems in use: <ul style="list-style-type: none"> • GPS: L1, L2, L5 • GLONASS: G1, G2 • Galileo: E5a, E5b, E5a+b, E6
NUM	Number of satellites used by each system
SNR	Quality of each SNR path, where: <ul style="list-style-type: none"> • A is > 20 dB • B is > 18 dB • C is > 15 db • D otherwise
RSF	Rover slip flag (non zero if parity errors in last 5 minutes, good for detecting jamming and TCXO issues)
BSF	Base slip flag
HAG	Horizontal accuracy guess
*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

Example \$PSAT,RTKSTAT,FIX,ROX,2,007F,9.5,L1,L2,G1,G2,9,5,0,0,A,A,B,B,0,1,
0.001*CC<CR><LF>

- L1 , L2 , G1 , G2 are the systems in use
- 9 satellites used by L1, 5 for L2, 0 for G1, 0 for G2
- A,A,B,B = quality of each SNR path
- RSF = 0
- BSF = 1
- HAG = 0.001

**Additional
Information**

**Related
Commands** [JASC,PSAT,RTKSTAT](#)

RD1 Message

Message Type [Data](#)

Description SBAS diagnostic information

Command Format to Request Message \$JASC , D1 , R[, OTHER] <CR><LF>

Message Format \$RD1 , SEC , WEEK , FREQ , DSPLOCK , BER2 , AGC , DDS , DOPPLER , DSPSTAT , ARMSTAT , DIFFSTAT , NAVCON*CC<CR><LF>

where:

Message Component	Description
SEC	Second of GPS week (may be a couple of seconds old)
WEEK	GPS week number
FREQ	L-band frequency in MHz (1575.4200 is used for SBAS)
DSPLOCK	N/A
BER2	BER - given for both SBAS satellites being tracked
AGC	L-band signal strength
DDS	0.0 for SBAS
DOPPLER	0 for SBAS
DSPSTAT	Status bit mask for the DSP tracking of SBAS <ul style="list-style-type: none"> • Bit 0 = Carrier lock • Bit 1 = BER OK (Viterbi lock) (yellow LED 2) • Bit 2 = L-Band: DSP got lock and has stable freq; WAAS: Frame sync2 • Bit 3 = Frame sync1 • Bit 4 = Track mode (same as carrier lock) • Bits 5 - 15 Unused
ARMSTAT	Status bit mask for the ARM GPS solution (ARM status values shown below) <ul style="list-style-type: none"> • Bit 0 = GPS lock (yellow LED 1) • Bit 1 = DGPS valid data • Bit 2 = ARM has lock • Bit 3 = Diff and GPS (flashing green LED 3) • Bit 4 = GPS solution is good (solid green LED 3) • Bit 5 = ARM controls yellow LED 2 • Bit 6 = ARM command for yellow LED 2 • Bits 7 - 15 Unused

DIFFSTAT	SBAS PRN of the satellite in use																								
NAVCON	<p>Series of hex character fields with each field representing the number of GPS satellites satisfying a certain condition, all of which conditions are required if the satellite is to be used in the solution</p> <p>Example of NAVCON for the value 179889A shown below (read right to left)</p> <table><tr><th><u>Hex Field</u></th><th><u>Description</u></th><th><u>Value</u></th></tr><tr><td>1 (right most field)</td><td>Hexadecimal count of satellites with valid tracks</td><td>A</td></tr><tr><td>2</td><td>Hexadecimal count of satellites for which an ephemeris message has been received</td><td>9</td></tr><tr><td>3</td><td>Hexadecimal count of satellites which are healthy</td><td>8</td></tr><tr><td>4</td><td>Hexadecimal count of satellites which passed the criteria of hex fields 1,2,3 and 5 (satellites that are tracked, have an ephemeris, are healthy, and are above the elevation mask)</td><td>8</td></tr><tr><td>5</td><td>Hexadecimal count of satellites above the elevation mask</td><td>9</td></tr><tr><td>6</td><td>Hexadecimal count of satellites for which a differential correction is available</td><td>7</td></tr><tr><td>7</td><td>Hexadecimal count of satellites for which a differential correction is NOT available</td><td>1</td></tr></table>	<u>Hex Field</u>	<u>Description</u>	<u>Value</u>	1 (right most field)	Hexadecimal count of satellites with valid tracks	A	2	Hexadecimal count of satellites for which an ephemeris message has been received	9	3	Hexadecimal count of satellites which are healthy	8	4	Hexadecimal count of satellites which passed the criteria of hex fields 1,2,3 and 5 (satellites that are tracked, have an ephemeris, are healthy, and are above the elevation mask)	8	5	Hexadecimal count of satellites above the elevation mask	9	6	Hexadecimal count of satellites for which a differential correction is available	7	7	Hexadecimal count of satellites for which a differential correction is NOT available	1
<u>Hex Field</u>	<u>Description</u>	<u>Value</u>																							
1 (right most field)	Hexadecimal count of satellites with valid tracks	A																							
2	Hexadecimal count of satellites for which an ephemeris message has been received	9																							
3	Hexadecimal count of satellites which are healthy	8																							
4	Hexadecimal count of satellites which passed the criteria of hex fields 1,2,3 and 5 (satellites that are tracked, have an ephemeris, are healthy, and are above the elevation mask)	8																							
5	Hexadecimal count of satellites above the elevation mask	9																							
6	Hexadecimal count of satellites for which a differential correction is available	7																							
7	Hexadecimal count of satellites for which a differential correction is NOT available	1																							
*CC	Checksum																								
<CR>	Carriage return																								
<LF>	Line feed																								

Additional Information

Related Commands [JASC.D1 \(RD1\)](#)

TSS1 Message

Message Type [Crescent Vector, Data](#)

Description Heading, pitch, roll, and heave message in the commonly used TSS1 message format

Command Format to Request Message \$JASC,PTSS1,R[,OTHER]<CR><LF>

where

- 'R' = message rate (in Hz) of 0 (off), 0.25, 0.5, 1, 2, 4, 5, 10, or 20 (if subscribed)
- ',OTHER' = optional field, enacts a change on the current port when you send the command without it (and without the brackets) and enacts a change on the other port when you send the command with it (without the brackets). See [Configuring the Data Message Output](#) for detailed information on 'THIS' and 'OTHER' port terminology.

Message Format :XXXXAASMHQHQMRRRRSMPPPP*CC<CR><LF>

where:

Message Component	Description						
XX	Horizontal acceleration						
AAAA	Vertical acceleration						
S	Space character						
M	Space if positive; minus if negative						
HHHH	Heave						
Q	Status flag <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>h</td><td>Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.</td></tr> <tr> <td>F</td><td>Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.</td></tr> </table>	Value	Description	h	Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.	F	Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.
Value	Description						
h	Heading aided mode (settling) - The System is receiving heading aiding signals from a gyrocompass but is still awaiting the end of the three minutes settling period after power-on or a change of mode or heave bandwidth. The gyrocompass may take several hours to settle after it has been powered-on. During this time, gyrocompass aiding of the System will not be perfect. The status flag does NOT indicate this condition.						
F	Full aided mode (settled condition) - The System is receiving and using aiding signals from a gyrocompass and from a GPS receiver or a Doppler log.						
M	Space if positive; minus if negative						
RRRR	Roll						
S	Space character						
M	Space if positive; minus if negative						
PPPP	Pitch						

*CC	Checksum
<CR>	Carriage return
<LF>	Line feed

**Additional
Information**

**Related
Commands** [JASC.PTSS1](#)

Resources

Reference Documents

National Marine Electronics Association
National Marine Electronics Association (NMEA) Standard for Interfacing Marine Electronic Devices
Version 2.1, October 15, NMEA 1995
7 Riggs Avenue
Severna Park, MD 21146
Tel: +1-410-975-9425
Tel Toll Free: +1-800-808-6632
<http://www.nmea.org/>

Radio Technical Commission for Maritime Services
RTCM Recommended Standards for Differential NAVSTAR GPS Service Version 2.2
Developed by Special Committee No. 104, RTCM 1998
1800 N Kent St, Suite 1060
Arlington, VA 22209, USA
Tel: +1-703-527-2000
<http://www.rtcn.org/>

Radio Technical Commission for Aeronautics
Minimum Operational Performance Standards (MOPS) for Global Positioning System/Wide Area Augmentation System Airborne Equipment
Document RTCA D0-229A, Special Committee No. 159, RTCA 1998
1828 L Street, NW, Suite 805
Washington, D.C. 20036 USA
Tel: +1-202-833-9339
<http://www.rtca.org/>

ARIC Research Corporation
Interface Control Document, Navstar GPS Space Segment/Navigation User Interfaces
ICD-GPS-200, April 12, 2000
2250 E. Imperial Highway, Suite 450
El Segundo, CA 90245-3509
<http://www.navcen.uscg.gov/>

Websites

Hemisphere GPS

<http://www.hemispheregps.com>

FAA WAAS

This site offers general information on the WAAS service provided by the U.S. FAAS.

http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/waas/

ESA EGNOS System Test Bed

This site contains information relating to past performance, real-time performance, and broadcast schedule of EGNOS.

<http://www.esa.int/esaNA/egnos.html>

Solar and Ionosphereic Activity

The following sites are useful in providing details regarding solar and ionospheric activity.

<http://iono.jpl.nasa.gov>

<http://www.spaceweather.com>

Troubleshooting

Use the following checklist to troubleshoot anomalous receiver system operation.

Receiver fails to power

- Verify polarity of power leads
- Check 1.0 A in-line power cable fuse
- Check integrity of power cable connections
- Check power input voltage
- Check current restrictions imposed by power source (minimum available should be > 1.0 A)

No data from receiver

- Check receiver power status
- Verify receiver is locked to a valid DGPS signal (this can often be done on the receiving device with the use of the PocketMAX PC)
- Verify receiver is locked to GPS satellites (this can often be done on the receiving device with the use of the PocketMAX PC)
- Check integrity and connectivity of power and data cable connections

Random data from receiver

- Verify the RTCM or the [Bin95](#) and [Bin96](#) messages are not being output accidentally (send a [JSHOW](#) command)
- Verify baud rate settings of receiver and remote device match correctly
- The volume of data requested to be output by the receiver potentially could be higher than the current rate supports. Try using 19200 or 38400 as the baud rate for all devices.

No GPS lock

- Check integrity of antenna cable
- Verify antenna has an unobstructed view of the sky
- Verify the lock status of the GPS satellites (this can often be done on the receiving device with the use of the PocketMAX PC)

No SBAS lock

- Check antenna connections
- Verify antenna has an unobstructed view of the sky
- Verify the lock status of SBAS satellites (this can often be done on the receiving device with the use of the PocketMAX PC - monitor BER value)

No DGPS position in external RTCM mode

- Verify that the baud rate of the RTCM input port matches the baud rate of the external source
- Verify the pinout between the RTCM source and the RTCM input port (transmit from the source must go to receiver of the RTCM input port and grounds must be connected)

- Verify the differential mode for the port which RTCM is being imported on is set to [JDIF,THIS](#)

Non-differential GPS output

- Verify receiver SBAS and lock status, or external source is locked

Multipath signals

- Operate away from large, reflective structures
- Use special antennas and GPS equipment to help reduce impact

Intermittent GPS Lock

- Check if receiver is installed in a noisy environment, and if so, shield the PCB
- Check that the antenna cable is a good quality cable and is not routed around digital or noisy components